

AFRL-VA-WP-TR-2001-3025

**DEVELOPMENT OF STRUCTURAL
INTEGRITY ANALYSIS METHODS
FOR AIRCRAFT STRUCTURES**

**AFGROW COMPONENT OBJECT MODEL (COM)
SERVER INTERFACE MANUAL, RELEASE 10**



Mr. James A. Harter

Mr. Alexander Litvinov

**Air Vehicles Directorate
2790 D Street, Street 504
Air Force Research Laboratory
WPAFB OH 45433-7542**

**Analytical Services & Materials, Inc.
107 Research Drive
Hampton, Virginia 23666**

MAY 2001

FINAL REPORT FOR THE PERIOD 01 MAY 1998 – 30 APRIL 2001

Approved for public release; distribution unlimited.

**AIR VEHICLES DIRECTORATE
AIR FORCE RESEARCH LABORATORY
AIR FORCE MATERIEL COMMAND
WRIGHT-PATTERSON AIR FORCE BASE, OH 45433-7542**

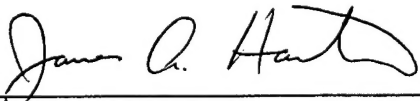
20020131 078

NOTICE

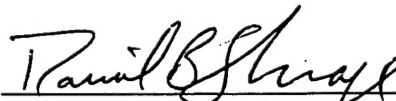
USING GOVERNMENT DRAWINGS, SPECIFICATIONS, OR OTHER DATA INCLUDED IN THIS DOCUMENT FOR ANY PURPOSE OTHER THAN GOVERNMENT PROCUREMENT DOES NOT IN ANY WAY OBLIGATE THE US GOVERNMENT. THE FACT THAT THE GOVERNMENT FORMULATED OR SUPPLIED THE DRAWINGS, SPECIFICATIONS, OR OTHER DATA DOES NOT LICENSE THE HOLDER OR ANY OTHER PERSON OR CORPORATION; OR CONVEY ANY RIGHTS OR PERMISSION TO MANUFACTURE, USE, OR SELL ANY PATENTED INVENTION THAT MAY RELATE TO THEM.

THIS REPORT IS RELEASABLE TO THE NATIONAL TECHNICAL INFORMATION SERVICE (NTIS). AT NTIS, IT WILL BE AVAILABLE TO THE GENERAL PUBLIC, INCLUDING FOREIGN NATIONS.

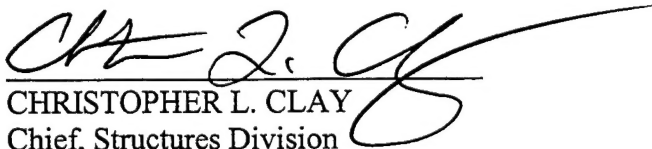
THIS TECHNICAL REPORT HAS BEEN REVIEWED AND IS APPROVED FOR PUBLICATION.



JAMES H. HARTER, Aero Engr
Analytical Structural Mechanics Branch
Structures Division
Air Vehicles Directorate



DANIEL B. SHRAGE, CAPT, USAF
Chief, Analytical Structural Mechanics Branch
Structures Division
Air Vehicles Directorate



CHRISTOPHER L. CLAY
Chief, Structures Division
Air Vehicles Directorate

Do not return copies of this report unless contractual obligations or notice on a specific document requires its return.

REPORT DOCUMENTATION PAGEForm Approved
OMB No. 074-0188

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing this collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188), Washington, DC 20503

1. AGENCY USE ONLY (Leave blank)		2. REPORT DATE MAY 2001	3. REPORT TYPE AND DATES COVERED Final, 05/01/1998 – 04/30/2001	
4. TITLE AND SUBTITLE DEVELOPMENT OF STRUCTURAL INTEGRITY ANALYSIS METHODS FOR AIRCRAFT STRUCTURES AFGROW COMPONENT OBJECT MODEL (COM) SERVER INTERFACE MANUAL, RELEASE 10			5. FUNDING NUMBERS C: F33615-94-D-3212 PE: 62201F PR: 2401 TA: LE WU: 00	
6. AUTHOR(S) James A. Harter (AFRL/VASM) Alexander Litvinov (Analytical Services & Materials, Inc.)				
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Air Vehicles Directorate 2790 D Street, Street 504 Air Force Research Laboratory WPAFB, OH 45433-7542			8. PERFORMING ORGANIZATION REPORT NUMBER Analytical Services & Materials, Inc. 107 Research Drive Hampton, Virginia 23666	
9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES) AIR VEHICLES DIRECTORATE AIR FORCE RESEARCH LABORATORY AIR FORCE MATERIEL COMMAND WRIGHT-PATTERSON AIR FORCE BASE, OH 45433-7542 POC: JAMES A. HARTER, AFRL/VASM, (937) 255-6104 x233			10. SPONSORING / MONITORING AGENCY REPORT NUMBER AFRL-VA-WP-TR-2001-3025	
11. SUPPLEMENTARY NOTES:				
12a. DISTRIBUTION / AVAILABILITY STATEMENT Approved for public release; distribution unlimited.			12b. DISTRIBUTION CODE	
13. ABSTRACT (Maximum 200 Words) The AFGROW COM Server allows users to perform automated crack growth life analyses using Microsoft Component Object Model (COM) technology. AFGROW may be called by any Windows software that is COM aware (Microsoft Office using Visual BASIC for Applications, Visual BASIC, Visual C++, Borland C++, and many others). This manual describes the use of the COM capabilities currently included in the crack growth life prediction software, AFGROW.				
14. SUBJECT TERMS Component Object Model, Crack Growth, Life Prediction, Fracture Mechanics, AFGROW			15. NUMBER OF PAGES 106	
			16. PRICE CODE	
17. SECURITY CLASSIFICATION OF REPORT Unclassified	18. SECURITY CLASSIFICATION OF THIS PAGE Unclassified	19. SECURITY CLASSIFICATION OF ABSTRACT Unclassified	20. LIMITATION OF ABSTRACT SAR	
NSN 7540-01-280-5500		Standard Form 298 (Rev. 2-89) Prescribed by ANSI Std. Z39-18 298-102		

TABLE OF CONTENTS

Section

LIST OF FIGURES	vii
FOREWORD	1
1. General Instructions	2
2. Using AFGROW with Visual BASIC for Applications	3
3. Application Object available properties and functions	7
3.1 AFGROW Application	7
3.1.1 Properties	7
3.1.2 Functions	7
AfgrowRetardationModels GetActiveRetardationModel()	7
short RunFrozPredict(double* Cycles, double* finalC, double* finalKc, double* finalA, double* finalKa, double* finalCt, double* finalKct)	7
boolean Quit ()	8
boolean RunPredict()	8
IRetardationInt* SetRetardation(AfgrowRetardationModels nModel, VARIANT dParameter, VARIANT bOpenLoadRatioAuto, VARIANT dOpenLoadRatio)	8
boolean SaveProblemDefFile(BSTR strFileName)	9
boolean OpenProblemDefFile(BSTR strFileName)	10
VARIANT CalculateBetas([in, optional] VARIANT aCrackLengthArray)	10
3.1.3 Events	12
Close()	12
FractureInfo(long nReason, double dStress, double dRStress, double dCycles, long dPass, IDispatch* iOutputInfo1, IDispatch* iOutputInfo2, IDispatch* iOutputInfo3, IDispatch* iOutputInfo4)	12
PredictInfo(AfgrowModels model, double dStress, double dRStress, double dCycles, long dPass, IDispatch* iOutputInfo1, IDispatch* iOutputInfo2, IDispatch* iOutputInfo3, IDispatch* iOutputInfo4)	12
PredictFinished(short result, double dblCycles, double dblKc, double dblKa, double dblKct, double dblC, double dblA, double dblCt)	13
TransitionInfo(long nType, long nReason, double dStress, double dRStress, double dCycles, long dPass, IDispatch* iOutputInfo1, IDispatch* iOutputInfo2, IDispatch* iOutputInfo3, IDispatch* iOutputInfo4)	14
3.2 AFGROW Model	15
3.2.1 Properties	15
3.2.2 Functions	16
boolean GetLoad(double* dTension, double* dBending, double* dBearing)	16
VARIANT GetWFStressDistr(CrackDirection nStressDir)	16
boolean IsCrackOffset()	17
boolean IsHoleOffset()	17
boolean IsObliqueCrack()	17
boolean NoCrackOffset()	17
boolean NoHoleOffset()	18
boolean NoObliqueCrack()	18
boolean SetCrackedPlateProp(double dWidth, double dThick)	18
boolean SetCrackedPlateWithHoleProp(double dWidth, double dThick, double dDia)	19
boolean SetCrackedPlateWithOffsetHoleProp(double dWidth, double dThick, double dDia, double dOffset)	19
SetCrackedPlateWithNotchProp(double dWidth, double dThick)	19
boolean SetCrkPltWithOffsetCrackProp(double dWidth, double dThick, double dCrackOffset)	20
boolean SetCylinderProp(double dWidth, double dInDia, double dOutDia)	20
boolean SetDiscProp(double dThick, double dDia)	20
short SetLoad(optional VARIANT dTension, optional VARIANT dBending, optional VARIANT dBearing)	21
boolean SetLugProp(double dWidth, double dThick, double dDia)	21

TABLE OF CONTENTS

Section

boolean SetObliqueCrack(double LengthC, double LengthCt)	21
boolean SetPipeProp(double dInDia, double dOutDia)	22
boolean SetRodProp(double dDia)	22
void SetStartPredictFromCycle(double dCycleNumber).....	22
boolean SetTransitionToOblique()	23
boolean SetWOLCTProp(double dWidth, double dThick, double dGrDepth, WOL_CTtype bType)	23
boolean SetWFStressDistr(CrackDirection nStressDir, VARIANT aStressDistr)	23
3.3 AFGROW Spectrum.....	24
3.2.1 Properties	24
3.3.2 Functions	24
boolean ConstAmplitudeSpectrum(double R)	24
boolean OpenSpectrumFile(BSTR sFileName).....	25
long GetSpectrumInfo (double* dCycles, double* dLevels, double* dSubspectra, double* dMaxValue, double* dMinValue)	25
3.4 AFGROW Materials	26
3.4.1 Functions	26
AfgrowMaterials GetActiveMaterial ()	26
IFormanMaterialInt* SetFormanMaterial ()	26
IHarterTMaterialInt* SetHarterTMaterial (optional VARIANT nMaterial)	26
INASGROMaterialInt* SetNASGROMaterial (optional VARIANT sMaterialName).....	27
ITabularLookupMaterialInt* SetTabularLookupMaterial ()	27
IWalkerMaterialInt* SetWalkerMaterial ()	28
3.5 Stress State.....	28
3.5.1 Functions	28
boolean GetStressState(double* dXDirValue, double* dYDirValue).....	28
boolean SetStressState(boolean bStressStateAuto, VARIANT dXDirValue, VARIANT dYDirValue)	28
4. Predict Preferences Object Available Parameters and Functions	30
4.1 Growth Increment Parameters	30
Properties	30
4.2 Output Options	30
Properties	30
4.3 Propagation Limits.....	30
4.3.1 Properties.....	30
4.3.2 Functions	30
SetPropagationLimits(optional VARIANT bCrackLengthStop, optional VARIANT bCyclesStop, optional VARIANT bKmaxStop, optional VARIANT bUserKmaxStop, optional VARIANT bNetSectionStop, optional VARIANT dCrackLength, optional VARIANT nCycleCount, optional VARIANT dUserKmax).....	30
4.4 Part Through to Through Crack Transition Options	31
Properties	31
4.5 Output Intervals	32
4.5.1 Properties.....	32
4.5.2 Functions	32
SetLifeInHours(boolean bLifeInHours, [in, optional] VARIANT dHoursPerPass);.....	32
SetOutputIntervals(OutputIntervalType nIntervalType, [in, optional] VARIANT aIntervalValue)	33
5. Beta Correction Object Available Parameters and Functions.....	34
Properties	34
VB sample	34
6. Residual Stress Object Available Parameters and Functions.....	36
Properties	36

TABLE OF CONTENTS

Section

VB sample	37
7. User Defined Beta Object Available Parameters and Functions	38
7.1 Properties	38
7.2 Functions	38
long GetFourPointsInt(double* dA12, double* dA34, double* dC1, double* dC2, double* dC3, double* dC4, double* dBetaA1, double* dBetaA2, double* dBetaA3, double* dBetaA4, double* dBetaC1, double* dBetaC2, double* dBetaC3, double* dBetaC4).....	38
long GetLineaInt(VARIANT* aLengthArray, VARIANT* aCLengthArray, VARIANT* aABetasArray, VARIANT* aCBetasArray).....	39
TypeOfUserDefBetaPartThroughCrackInt GetPartThroughIntType ()	40
boolean IsSet().....	40
long SetFourPointsInt(double dA12, double dA34, double dC1, double dC2, double dC3, double dC4, double dBetaA1, double dBetaA2, double dBetaA3, double dBetaA4, double dBetaC1, double dBetaC2, double dBetaC3, double dBetaC4)	41
long SetLineaInt(VARIANT aLengthArray, VARIANT aCLengthArray, VARIANT aABetasArray, VARIANT aCBetasArray).....	41
8. Harter-T Material Object Available Parameters and Functions	44
Functions	44
boolean IsActive ()	44
boolean SetActive(VARIANT nMaterial)	44
9. Walker Material Object Available Parameters and Functions	45
9.1 Properties	45
9.2 Functions	45
boolean IsActive ()	45
boolean SetActive()	45
10. Forman Material Object Available Parameters and Functions	46
10.1 Properties	46
10.2 Functions	46
boolean IsActive ()	46
boolean GetMapR([out] double* dRloBorder, [out] double* dRhiBorder,[out] double* dRValue)	46
boolean GetUserRRange(double* dRValue, VARIANT* aCurveSegments).....	47
boolean SetActive()	48
long SetMapR(boolean bMapR, optional VARIANT dRloBorder, optional VARIANT dRhiBorder, optional VARIANT dRValue).....	48
long SetUserRRange(boolean bUserRRange, VARIANT dRValue, VARIANT aCurveSegments)	49
11. TabularLookup Material Object Available Parameters and Functions	50
11.1 Properties	50
11.2 Functions	50
boolean IsActive ()	50
boolean SetActive()	50
boolean SetData(VARIANT aDataMatrix, double dRlo, double dRhi, double ddadNlo, double ddadNhi, double dThreshold)	51
12. NASGRO Material Object Available Parameters and Functions	52
Functions	52
boolean IsActive ()	52
boolean SetActive(VARIANT sMaterialName).....	52
13. Crack Initiation Object Available Parameters and Functions	53
13.1 Properties	53
13.2 Functions	53
DataType GetCyclicDataType ().....	53
DataType Get GetStrainLifeDataType Type ()	53
DataType SetCyclicDataType (DataType nDataType, [in, optional] VARIANT aData).....	54
DataType SetStrainLifeDataType (DataType nDataType, [in, optional] VARIANT aData).....	54

TABLE OF CONTENTS

Section

14. Retardation Object Available Parameters and Functions	54
14.1 Properties	54
14.2 Functions	55
AfgrowRetardationModels GetActiveRetardationModel()	55
long GetClosureModelParam(double* dParameter, boolean* bOpenLoadRatioAuto, double* dOpenLoadRatio)	55
AfgrowRetardationModels NoRetardation()	55
AfgrowRetardationModels SetClosureModel(VARIANT dParameter, VARIANT bOpenLoadRatioAuto, VARIANT dOpenLoadRatio)	56
AfgrowRetardationModels SetWheelerModel(VARIANT dParameter)	56
AfgrowRetardationModels SetWillenborgModel(VARIANT dParameter)	56
15. Output Object Parameters	58
15.1 Properties	58
15.2 VB sample	58
Appendix 1: Return Codes	59
Application Object - Predict Function Return Codes	59
Application Object - SetLoad Function Return Codes	59
User-Defined Beta Object - Interpolation Function Return Codes	60
Forman Material Object - Function Return Codes	60
Application Object - Return Codes for Transition Types	61
Application Object - Return Codes for Transition Types	61
Application Object - Return Codes for Fracture Types	61
Application Object -Return Codes for GetSpectrumInfo function	61
Appendix 2: AFGROW Constants	62
AFGROW Models (AfgrowModels)	62
Common Crack Growth Specimen Types (WOL_CTtype)	62
Types of Units (AfgrowUnits)	63
Crack Growth Directions (CrackDirection)	63
Crack Transition Conditions (CrackTransConditions)	63
Types of Beta Correction Data (TypeOfBetaCorData)	63
Types of Residual Stress Data (CrackDirection)	63
Residual Stress SIF Table Generation Tyoes (TypeOfResSIFTTableGeneration)	63
Types of Material Data Available in AFGROW (AfgrowMaterials)	63
Types of SIF Data (DataType)	63
Types of User-Defined Beta Interpolation (Part-Through Cracks) (TypeOfUserDefBetaPartThroughCrackInt)	64
Types of Retardation Models (AfgrowRetardationModels)	64
Types of Output Intervals (OutputIntervalType)	64
Appendix 3: AFGROW Exceptions	65
Appendix 4: Sample VBA Source Code (without Events)	66
Appendix 5: Sample VBA Source Code (with Events)	67
Appendix 6: Sample VBA Source Code (Using Predict Preferences)	68
Appendix 7: Sample VBA Source Code (Using Beta Correction Tables)	71
Appendix 8: Sample VBA Source Code (User-Defined Betas and Tabular Crack Growth Rate)	74
Appendix 9: Sample VBA Source Code (Using Predict Info)	78
Appendix 10: Material Codes	82
Harter T Material Data Codes	82
NASGRO Material Data Codes	82

LIST OF FIGURES

Figure

1: VBA Tools, Reference Dialog Box.....	3
2: Sample Interface Dialog	4
3: AFGROW COM Server Control Menu	6
4: Four Point Interpolation Page of the User-Defined Beta Wizard	38

FOREWORD

This report was originally prepared by Analytical Services & Materials, Inc., Hampton Virginia for AFRL/VASE, Wright-Patterson Air Force Base, Ohio under contract F33615-94-D-3212 "Development of Structural Integrity Analysis and Verification for Aircraft Structures." The government contract manager was Mr. James A. Harter, AFRL/VASM.

The work performed under this report (Delivery Order 0011) was performed by Analytical Services & Materials, Inc. personnel located at the AFRL/VASM Fatigue and Fracture Test Facility, Bldg. 65, Area B. The authors of this report were Mr. Alexander V. Litvinov and Mr. James A. Harter.

1. General Instructions

NOTE: This capability requires a certain amount of programming knowledge and may not be something that everyone will want to try. Please read the all of the information provided in the following sections before attempting to use this feature.

The server version of AFGROW for Windows 95/98/NT4/ME/2000® may be used interactively (as always) or by other windows programs. Microsoft's Component Object Model (COM) Server technology has been added to AFGROW to allow users to perform multiple crack growth analyses and have the results sent directly to the calling program. Any Windows compatible application development software may be used (as long as it supports the Microsoft COM standard). These development platforms include: Visual BASIC 5.0, Visual Basic for Applications (Office 97), Visual C++ 5.0, and Borland C++ Builder 4. Visual BASIC for Applications (VBA) is probably the most widely available platform, which can be used for this purpose.

Before using the server from another windows program, AFGROW MUST be executed at least once as a stand-alone program. When the server version is executed for the first time, Windows will recognize that it is a COM server and will look for a Type Library Binary (TLB) file (*afgrow.tlb*) and register AFGROW as a COM object on the local machine. Once this is complete, the AFGROW server will be available for use by other COM compatible software.

The TLB file contains detailed information that other programs use to determine which variables and subroutines are available. When the AFGROW server is updated and a new version is downloaded, all references to the previous server version MUST be updated.

The AFGROW COM server consists of several interfaces, each with a specific purpose. There are too many options available in AFGROW to use a single interface. The details for each interface are given in sections 3 through 15. A Microsoft COM interface is the vehicle through which COM objects are accessed. An interface is a collection of functions and variables logically grouped together. It is analogous to file systems and directories on a hard disk. It would not make sense to put all of the files on a hard disk in a single directory. Interfaces are used to help organize the COM server.

Each AFGROW COM interface includes functions that allow several variables to be set at one time. These functions should be used in most cases to avoid errors when variables are interdependent. Individual variables may be set independently, but users must be sure to understand how each variable is used.

Again, remember that AFGROW will still function as a stand-alone interactive code as it has in the past. The COM capability is merely an addition to AFGROW, which we hope is useful.

It is strongly recommended that users read this document, in its entirety, prior to using the COM capabilities of AFGROW.

2. Using AFGROW with Visual BASIC for Applications

Visual BASIC for Applications is a version of BASIC that Microsoft distributes with its Office software suite and is generally used to manipulate data in Excel and Access. The following discussion will address the use of VBA in Excel.

First, open Excel and use the control toolbar to create any desired controls to be used in the VBA program (i.e., a pushbutton control). After the control(s) are established, select the view code button on the control toolbar. At this point, a new VBA window will open showing a blank subroutine for the control you have created. When creating a new program, always verify that the AFGROW server is referenced by selecting the **tools, references** menu item. The reference window will be opened (figure 1).

Refer to the sample source code provided in appendix 4 to establish the reference to the AFGROW server application interface (*Dim mAfgrow As afgrow.Application*) and create the afgrow object (*Set mAfgrow = CreateObject("afgrow.application")*).

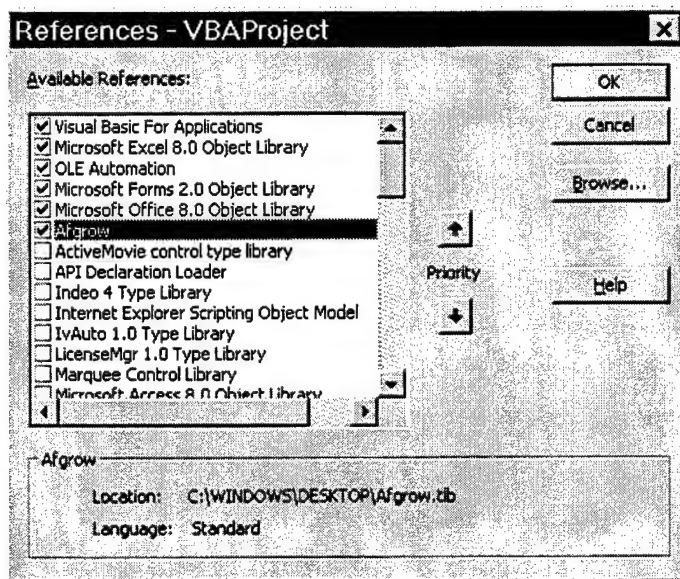


Figure 1: VBA Tools, Reference Dialog Box

In the sample, references to AFGROW will have the following format *mAfgrow.****. The reference variable name doesn't have to be *mAfgrow*, it can be any name the user chooses. VBA offers quite a bit of real-time help while the code is being written. When the period in the afgrow reference is typed, VBA automatically opens a dialog box as follows:


```

i = 3
Set mAfgrow = CreateObject("Afgrow.Application")
Do While Sheet1.Cells(i, 2) > 0
    mAfgrow.Model = Sheet1.Cells(i, 6)
    mAfgrow.|
    mAfgrow.BetaCorrection Cells(i, 3))
    mAfgrow.ConstAmplitudeSpectrum 4)
    mAfgrow.CrackInitiation 5)
    mAfgrow.CrackLengthA
    result.CrackLengthACConst finalC, finalKc, finalA, finalKt
    Sheet1.CrackLengthC
    Sheet1.CrackLengthCt
    Sheet1.Cells(i + 14, 2) = Cycles
    Sheet1.Cells(i + 14, 3) = mAfgrow.SME
    i = i + 1
Loop
mAfgrow.Quit

```

Figure 2: Sample Interface Dialog

The dialog box above lists all of the variables and subroutines available in the AFGROW server application interface. Simply double-click on any selection and the name will automatically be appended to the reference name. If a function is selected, a list of variables and the value(s) to be returned from the subroutine will be indicated as soon as an empty space is entered after the subroutine name. Remember that, in VBA, when the parameters passed to and from a function are grouped inside parentheses VBA expects the function to return a value. If the function DOES NOT return a value, do not use parentheses – just use an empty space followed by the parameter list. A complete list of all variables and subroutines available in the AFGROW server may be found in the VBA window by selecting the view, object browser menu (look at the applications listed).

Finally, note that there are two ways to execute a life a prediction using AFGROW as a COM server. The first way (*RunFrozPredict*) is used in the example since it returns several variables and does not require the code to receive messages from AFGROW. The subroutine is referred to as frozen since the calling application will be frozen (unable to respond to user requests) until the prediction is complete. The other way (*RunPredict*) to execute AFGROW requires the user to receive messages (or events) from AFGROW.

The first method may be good for relatively simple cases where limited information is needed. The latter method (*RunPredict*) is much more flexible and is recommended for most practical uses.

The example with events (or messages - see appendix 5) is written very differently than the example without events. There is no need to declare (or dimension) the AFGROW output variables since this job is done by VBA in the *mAfgrow_PredictFinished* call. VBA constructs the subroutine automatically after the line: *Public WithEvents mAfgrow As Afgrow.Application* is entered. After entering the line, select afgrow (or whatever name you have chosen) on the left-hand combo box above the window where the program is displayed. After the application name is selected, a list of available routines will be displayed on the right-hand combo box. Select *PredictFinished*, and the routine will be added to the code. The other difference is that the code must wait for the *PredictFinished* event before the next run can be started. Do not perform the *RunPredict* routine inside a while loop - the code will not be able to detect the *PredictFinished* event because it would be busy executing the loop. In the example code, the first call to *RunPredict* is made from the initial button_click, but the subsequent calls are made from the *PredictFinished* routine which is triggered after the *PredictFinished* event is detected.

Any time a call to RunPredict is made, the code should be waiting for an event, and therefore not be executing other commands at the same time.

The example using the Predict Preferences interface (see appendix 6) adds the ability to set limits to the crack growth life prediction and allows output to be directed to an standard text data file (which may be printed) or an ASCII plot file. A new line must be added to the code after referencing the AFGROW application (*Public prefs As PredictPreferences*). This line doesn't require the prefix *mAfgrow* since there are no other *PredictPreferences* classes available (look at the object browser). Before using the preference items, you must add the line: *Set prefs = mAfgrow.PredictPreferences*. In the example, the following line: *Call prefs.SetPropagationLimits(True, , , , 1.5)* uses call to transfer control to the function (*SetPropagationLimits*) . If call is used, any value returned by the function will be discarded. VBA will provide a help dialog for functions that will indicate which variables belong in the function and in which order. In the case above, users can stop entering commas after the last value being used.

The example using the beta correction table interface (see appendix 7) adds the ability to enter beta correction information (stresses or betas). A new line must be added to the code after referencing the AFGROW application (*Public betacorr As BetaCorrection*). This line doesn't require the prefix *mAfgrow* since there are no other *BetaCorrection* classes available (look at the object browser). If residual stresses are required, the same format is used, but the *ResidualStress* object must be referenced. Before using the "*BetaCorrection*" or "*ResidualStress*" items, user must add a line of code to create an object that can be used in the client software. In the example, the beta correction object is created as follows: *Set betacorr = mAfgrow.BetaCorrection*. It is also important to note that the model should be set before data are entered since a part-through crack requires values in two directions and a through crack requires values in only one direction. Refer to the detailed help in sections 5 and 6 in this document for more important information. The data type for the arrays are variant and the variable, *middle*, in the example is required to allow the array of double precision to be set equal to the variant array used in the server (refer to the example). There may be more elegant ways to accomplish this, but the above method works.

The example with user-defined betas and tabular lookup crack growth rate interfaces (see Appendix 8) is similar to the previous example. It also uses the standard application interface and the predict preferences interface. This example uses multiple Excel worksheets to organize the crack growth rate data and user-defined beta values. The details for adding user-defined betas and material data to any COM client are given in sections 7 through 12.

The capability to record crack growth rate data for a given analysis (*PredictInfo* event) is covered in section 3.1.3 (events) and section 15 (see appendix 9). This example builds on the previous example by adding the ability to record cycles, crack length, beta values for a given crack growth analysis. This may be useful in cases where automated plots of certain parameters are desired for a given analysis.

This AFGROW COM server has also added an icon that appears in the Windows taskbar tray (the taskbar is where the start button is located, and the tray is on the right-hand side of the taskbar). When the server is running, the AFGROW icon will appear in the tray. The server may be controlled by right-clicking on the AFGROW icon. The following menu will appear.

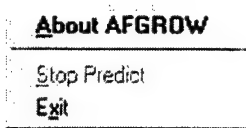


Figure 3: AFGROW COM Server Control Menu

The server may be stopped, closed (exit), or version information may be obtained.

Users can also create forms using VBA. The option to create a form allows the user to create a custom interface where information can be entered or edited. Everything else is the same - just use the information above to write the VBA code.

Notes:

If the AFGROW server does not work after installing a version upgrade, check to see whether or not there are multiple copies of a file named `afgrow.tlb`. This file contains information about the COM server options for AFGROW. This file is placed in the AFGROW directory. There should only be a single `afgrow.tlb` file on a machine. Also, always remember to remove the previous version of AFGROW before loading a new version. Use the control panel option to add/remove software to remove the old version. If there are still problems running an old VBA code with a newer version of AFGROW, try moving the new TLB file to a floppy (remove it entirely from the computer), and run the VBA code. Make sure there are no other `afgrow.tlb` files on the computer. Close the VBA code and replace the TLB file in the AFGROW directory and run AFGROW (as an interactive program) to re-register the TLB file. Next, open the VBA code and VBA editor, select tools, references and browse for the TLB file.

Never attempt to use empty spreadsheet cells to fill values required by the COM server. Empty cells are not equivalent to zero.

When a VARIANT type is set equal to array, it is important that the array be properly dimensioned. If the array is larger than required, extra values will be included in the VARIANT causing problems with the function using the VARIANT. If the array is too small, required variable cannot be saved. Remember, that in Visual Basic if array declared as `test[3]` it contains four values (`test[0]`, `test[1]`, `test[2]` and `test[3]`). Also, when setting VARIANT values with Excel cell values, the cell values need to be cast to the appropriate type (integer, double, etc.).

3. Application Object available properties and functions

3.1 AFGROW Application

3.1.1 Properties

Name	Type	Description
BetaCorrection	IBetaCorrectionInt*	Returns the Beta Correction AFGROW object (see section 5). Read-only
CrackInitiation	IcrackInitiation*	Returns Crack Initiation AFGROW object (see section 12). Read-only
ResidualStresses	IResidualStressInt*	Returns the Residual Stresses AFGROW object (See Section 6). Read-only
PredictParameters	IPredictPropertyInt*	Returns the Predict Properties AFGROW object (see section 4). Read-only
Visible	Boolean	TRUE if the AFGROW is visible, and FALSE if not.
Units	AfgrowUnits	Active AFGROW Units enumerated in AfgrowUnits data type (see appendix 2).
UserDefinedBeta	IUserDefBetaInt*	Returns User Defined Beta AFGROW object (see section 7). Read-only

3.1.2 Functions

AfgrowRetardationModels GetActiveRetardationModel()

Description

Gets the Active (Selected) in AFGROW retardation model.

Returning value

Returns the Active (Selected) in AFGROW retardation model. AFGROW retardation models enumerated in AfgrowRetardationModels data type (see appendix 2).

short RunFrozPredict(double* Cycles, double* finalC, double* finalKc, double* finalA, double* finalKa, double* finalCt, double* finalKct)

(Ignore pointers () in Visual Basic)*

Description

The subroutine is referred to as frozen since the calling application will be frozen (unable to respond to user requests) until the prediction is complete. The other way (*RunPredict*) to execute AFGROW requires the user to receive messages (or events) from AFGROW.

Returning value

A **short** value that specifies the returning result of the function. If it is not equal to 0, the function returns an error code. All error codes are provided in appendix 1.

Parameters

Name	Type	Description
Cycles	double*	Number of cycles to failure
FinalC	double*	Final crack length in C (X) direction
FinalKc	double*	Final K in C (X) direction
FinalA	double*	Final crack length in A (Y) direction
FinalKc	double*	Final K in A (Y) direction
FinalCt	double*	Final crack length in Ct (short side of the oblique through crack in X direction)
FinalKct	double*	Final K in Ct (X) direction

boolean Quit ()

Description

Quits AFGROW, prepares AFGROW object for uploading. Use only in case when AFGROW object was created by a client application.

Returning value

Return TRUE if function execution was successful; otherwise, FALSE.

boolean RunPredict()

Description

Execute the predict routine in AFGROW in an interactive mode. Requires the user to receive messages (or events) from AFGROW.

Returning value

Returns TRUE if function execution was successful; otherwise, FALSE.

IRetardationInt* SetRetardation(AfgrowRetardationModels nModel, VARIANT dParameter, VARIANT bOpenLoadRatioAuto, VARIANT dOpenLoadRatio)

Description

Sets the Retardation Model in AFGROW (see section 14).

Returning value

Retardation Model AFGROW Object.

Parameters

Name	Type	Description
nModel	AfgrowRetardationModels	Active (selected) retardation model. AFGROW retardation models are enumerated in the AfgrowRetardationModels data type (see appendix 2).
dParameter	VARIANT ¹	Retardation parameter. Shut off Ratio for Willenborg model, Open Load Ratio for Closure model, and Wheeler shaping exponent for Wheeler model. This parameter is ignored by AFGROW when the user selects no retardation (see description of AfgrowRetardationModels data type in appendix 2).
bOpenLoadRatioAuto	VARIANT	Used to specify initial crack opening level for Closure Retardation model and will be Ignored by AFGROW in other cases. If this parameter is TRUE, program uses the first cycle of the input spectrum to determine the Open Load Ratio parameter. If it is FALSE, AFGROW expects users to set dOpenLoadRatio parameter in this function and it will be used as the initial crack opening level.
dOpenLoadRatio	VARIANT	Initial crack opening level. Used only by the Closure model when bOpenLoadRatioAuto is FALSE.

boolean SaveProblemDefFile(BSTR strFileName)

Description

Save the current AFGROW problem definition file.

Returning value

Return TRUE if function execution was successful; otherwise, FALSE.

Parameters

Name	Type	Description
StrFileName	BSTR	Full name of the AFGROW problem definition file

¹ The Variant data type is the data type for all variables that are not explicitly declared as some other type (using statements such as Dim, Private, Public, or Static). The Variant data type has no type-declaration character.

For more on Variant (VB) or VARIANT (VC++), look in help provided with this languages.

		(*da3)
--	--	--------

boolean OpenProblemDefFile(BSTR strFileName)

Description

Open a new AFGROW problem definition file.

Returning value

Return TRUE if function execution was successful; otherwise, FALSE.

Parameters

Name	Type	Description
StrFileName	BSTR	Full name of the AFGROW problem definition file (*da3)

VARIANT CalculateBetas([in, optional] VARIANT aCrackLengthArray)

Description

Get the beta factors at the user specified crack lengths for the currently selected model. Remember that AFGROW defines beta as: $\text{Beta} = K/(\text{Stress} \times \text{SQRT}(\text{PI} \times \text{crack length}))$.

Returning value

VARIANT (1..2) Dimensional array filled with the beta values for the crack directions (C, A or Ct). For all through cracks AFROW returns 1-Dimensional array, for oblique and part through cracks, a 2-Dimensional array is returned. Each row of the array contains beta values that correspond to the passed crack lengths. Maximum allowed size of the array is 25x2.

Parameters

Name	Type	Description
aCrackLengthArray	VARIANT	(1..2) Dimensional array filled with the required crack lengths for the crack directions C and A (or Ct as the 2 nd dimension for an oblique case). For all through cracks AFROW expects 1-Dimensional array, for oblique and part through cracks 2-Dimensional array. If this parameter omitted, AFGROW will calculate betas for currently selected crack parameters.

VB example

```

Private Sub bBetaCalc_Click()
Dim aBetas As Variant
Dim aCracks(2, 1) As Double
Dim error As Integer

aCracks(0, 0) = 0.03
aCracks(0, 1) = 0.03
aCracks(1, 0) = 0.05
aCracks(1, 1) = 0.07
aCracks(2, 0) = 0.11
aCracks(2, 1) = 0.08

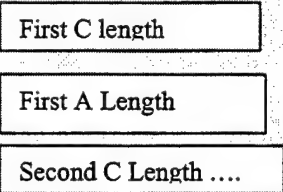
mAfgrow.Model = aSingleThroughAtHole
error = mAfgrow.SetCrackedPlateWithOffsetHoleProp(4, 0.25, 0.25, 0.5)
If (error = False) Then MsgBox (" Error Setting Model Prop")

mAfgrow.SetObliqueCrack 0.15, 0.09

On Error GoTo error
aBetas = mAfgrow.CalculateBetas(aCracks)
On Error Resume Next
txtBetaValue.Text = aBetas(0)
If error <> 0 Then
txtBetaValue.Text = aBetas(0, 0) & ", " & aBetas(0, 1)
End If
Exit Sub
error:
If Err.Number <> 0 Then MsgBox "Message from " + Err.Source + " : " +
Err.Description

End Sub

```



3.1.3 Events

Close()

Description

Occurs before AFGROW closes. Tells the client application that the server is closing.

FractureInfo(long nReason, double dStress, double dRStress, double dCycles, long dPass, IDispatch* iOutputInfo1, IDispatch* iOutputInfo2, IDispatch* iOutputInfo3, IDispatch* iOutputInfo4)

Description

Called when fracture occurs during AFGROW execution of the prediction routine.

Parameters

Name	Type	Description
nReason	long	Value that specifies the code for the reason that fracture occurred. All fracture codes are provided in appendix 1.
dStress	double	Current stress value
dRStress	double	Current stress ratio
dCycles	double	Number of cycles or life in hours (depends on AFGROW preferences)
dPass	long	Number of passes through the spectrum
iOutputInfo1	IDispatch*	Output information for the first AFGROW output (C (X) or C1 (X)) crack direction (see section 15 for a list of available parameters)
iOutputInfo2	IDispatch*	Output information for the second output (A (Y), Ct (X), or C2(X)) crack direction. It depends on the active AFGROW model. May also be equal to NULL (Empty in VB) (see section 15 for a list of available parameters).
iOutputInfo3	IDispatch*	Reserved value, always equal to NULL.
iOutputInfo4	IDispatch*	Reserved value, always equal to NULL.

PredictInfo(AfgrowModels model, double dStress, double dRStress, double dCycles, long dPass, IDispatch* iOutputInfo1, IDispatch* iOutputInfo2, IDispatch* iOutputInfo3, IDispatch* iOutputInfo4)

Description

Contains output information and is called at each AFGROW output interval.

Parameters

Name	Type	Description
model	AfgrowModels	Active AFGROW model. The AFGROW Model identification number is enumerated in AfgrowModels (see appendix 2).
dStress	double	Current stress value
dRStress	double	Current stress ratio
dCycles	double	Number of cycles or life in hours (depends on AFGROW preferences)
dPass	long	Number of passes through spectrum
iOutputInfo1	IDispatch*	Output information for the first AFGROW output (C (X) or C1 (X)) crack direction (See Section 15 for a list of available parameters)
iOutputInfo2	IDispatch*	Output information for the second output (A (Y), Ct (X), or C2 (X)) crack direction. It depends on the active AFGROW model. May also be equal to NULL (Empty in VB). (see section 15 for a list of available parameters)
iOutputInfo3	IDispatch*	Reserved value, always equal to NULL.
iOutputInfo4	IDispatch*	Reserved value, always equal to NULL.

PredictFinished(short result, double dblCycles, double dblKc, double dblKa, double dblKct, double dblC, double dblA, double dblCt)

Description

Occurs when AFGROW finishes execution of the prediction routine. **Note:** This event does not necessarily indicate that fracture has occurred – only that the prediction routine finished for some reason.

Parameters

Name	Type	Description
Result	short	Value that specifies the return code of the function. If it is not equal to 0, the function finished the prediction and the returning result contains an error code. All error codes are provided in appendix 1.
Cycles	double	Number of cycles to failure
FinalC	double	Final crack length in the first (C (X) or C1 (X)) direction
FinalKc	double	Final K in the first (C (X) or C1 (X)) direction
FinalA	double	Final crack length in the second (A (Y) or C2 (X)) direction
FinalKc	double	Final K in the second (A (Y) or C2 (X)) direction
FinalCt	double	Final crack length in Ct (short side of the oblique through crack in X direction)
FinalKct	double	Final K in Ct (X) direction

TransitionInfo(long nType, long nReason, double dStress, double dRStress, double dCycles, long dPass, IDispatch* iOutputInfo1, IDispatch* iOutputInfo2, IDispatch* iOutputInfo3, IDispatch* iOutputInfo4)

Description

Called when transition from one type of the crack to another occurs during AFGROW execution of the prediction routine.

Parameters

Name	Type	Description
nType	long	Value that specifies the transition type. All transition type codes are provided in appendix 1.
nReason	long	Value that specifies the code specifying the reason for transition. All transition codes are provided in appendix 1.
dStress	double	Current stress value
dRStress	double	Current stress ratio
dCycles	double	Number of cycles or life in hours (depends on AFGROW preferences)
dPass	long	Number of passes through spectrum
iOutputInfo1	IDispatch*	Output information for the first AFGROW output (C (X) or C1 (X)) crack direction (see section 15 for a list of available parameters)
iOutputInfo2	IDispatch*	Output information for the second output (A (Y), Ct (X), or C2 (X)) crack direction. It depends on the active AFGROW model. May also be equal to NULL (Empty in VB) (see section 15 for a list of available parameters).
iOutputInfo3	IDispatch*	Reserved value, always equal to NULL.
iOutputInfo4	IDispatch*	Reserved value, always equal to NULL.

3.2 AFGROW Model

3.2.1 Properties

Name	Type	Description
bInPlaneConstraint	boolean	Prevent in-plane bending (only for edge cracks)
CrackLengthA	double	Crack length in A (Y) direction in AFGROW
CrackLengthC	double	Crack length in C (X) direction in AFGROW
CrackLengthCt	double	Short side of the oblique crack length in the C (X) direction in AFGROW
CrackLengthACConst	boolean	Keep constant A/C length ratio in AFGROW (Controlled by the C-direction)
FilledUnloadedHole	boolean	Filled, unloaded hole for Cracked Plate with Hole specimens
Model	AfgrowModels	AFGROW Model identification number - enumerated in AfgrowModels (see appendix 2)
SpecimenCrackOffset	double	Crack offset (Non centered crack - valid only for Cracked Plate with internal through crack model)
SpecimenCylInDia	double	Inner diameter of the Cylinder (Thick Pipe) specimen ¹
SpecimenCylOutDia	double	Outer diameter of the Cylinder (Thick Pipe) specimen ¹
SpecimenDiscDia	double	Diameter of the Disk specimen
SpecimenHoleDia	double	Hole diameter of the Cracked Plate with Hole or Lug specimen
SpecimenHoleOffset	double	Offset of the hole in Cracked Plate with Hole specimen
SpecimenGrooveDepth	double	Groove Depth of the Common Crack Growth specimen (CT or WOL)
SpecimenPipeInDia	double	Pipe inner diameter ²
SpecimenPipeOutDia	double	Pipe outer diameter ¹
SpecimenRodDia	double	Rod diameter
SpecimenThickness	double	Thickness of the specimen
SpecimenWidth	double	Width of the specimen
SpecimenWOLSTType	WOL_CTtype	Type of the Common Crack Growth specimen enumerated in WOL_CTtype (see appendix 2)

¹ Applies to Internal and External Axial Crack in Thick Pipe (Weight Function) solutions.

² Applies to Part-Through and Through Crack in Pipe (Standard) solution

3.2.2 Functions

boolean GetLoad(double* dTension, double* dBending, double* dBearing)

(Ignore pointers () in Visual Basic)*

Description

Get the ratio of the tension, bending, or bearing stress to the reference stress for the load case being modeled.

Returning value

Return TRUE if function execution is successful; otherwise, FALSE.

Parameters

Name	Type	Description
dTension	double*	ratio of the tension stress to the reference stress
dBending	double*	ratio of the bending stress to the reference stress
dBearing	double*	ratio of the bearing stress to the reference stress

VARIANT GetWFStressDistr(CrackDirection nStressDir)

Description

Get the applied stress distribution for models using the Weight Function Stress Intensity Solution.

Returning value

VARIANT 2-Dimensional array filled with stress distribution values for the direction defined by **nStressDir** variable. Each row of the array contains the distance from the crack origin and corresponding stress values. Maximum allowed size of the array is 2x25.

Parameters

Name	Type	Description
nStressDir	CrackDirection	AFGROW crack growth direction enumerated in the CrackDirection data type (see appendix 2).

VB example

```
Dim SDXarray As Variant
Dim lvalue As Integer
Dim fvalue As Integer
SDXarray = mAfgrow.GetWFPStressDistr(0)
If IsArray(SDXarray) Then
    lvalue = UBound(SDXarray, 1)
    fvalue = LBound(SDXarray, 1)
    MsgBox "Last X=" & CStr(SDXarray(lvalue, 0)) & " Last Stress="
    & CStr(SDXarray(lvalue, 1))
End If
```

boolean IsCrackOffset()

Description

Determine whether the Crack in the Cracked Plate is offset (non centered).

Returning value

Return TRUE if the crack is offset; otherwise, FALSE.

boolean IsHoleOffset()

Description

Check if the Hole in the Cracked Plate with Hole Specimen is offset (non centered).

Returning value

Return TRUE if the hole is offset; otherwise, FALSE.

boolean IsObliqueCrack()

Description

Check whether the Through Crack in the Cracked Plate with Hole Specimen is oblique. If the crack is Part-Through, check whether the crack will transition to an Oblique Through Crack.

Returning value

Return TRUE if the crack is oblique; otherwise, FALSE.

boolean NoCrackOffset()

Description

Set no offset for the crack in the Cracked Plate Specimen (the crack will be centered).

Returning value

Return the previous offset state.

boolean NoHoleOffset()

Description

Set no offset hole for the Cracked Plate with Hole Specimen (the hole will be centered).

Returning value

Return the previous offset state.

boolean NoObliqueCrack()

Description

Set no oblique through crack for the Cracked Plate with Hole Specimen. If the crack is Part-Through, it will transition to a non oblique (Straight) Through Crack.

Returning value

Return the previous Oblique Crack state.

boolean SetCrackedPlateProp(double dWidth, double dThick)

Description

Sets all Cracked Plate Specimen properties.

Returning value

Return TRUE if function execution was successful; otherwise, FALSE.

Parameters

Name	Type	Description
dWidth	double	Width of the specimen
dThick	double	Thickness of the specimen

boolean SetCrackedPlateWithHoleProp(double dWidth, double dThick, double dDia)

Description

Set all Cracked Plate with Hole Specimen properties.

Returning value

Return TRUE if function execution was successful; otherwise, FALSE.

Parameters

Name	Type	Description
dWidth	double	Width of the specimen
dThick	double	Thickness of the specimen
dDia	double	Hole diameter of the specimen

boolean SetCrackedPlateWithOffsetHoleProp(double dWidth, double dThick, double dDia, double dOffset)

Description

Set all Cracked Plate with Offset Hole Specimen properties.

Returning value

Return TRUE if function execution was successful; otherwise, FALSE.

Parameters

Name	Type	Description
dWidth	double	Width of the specimen
dThick	double	Thickness of the specimen
dDia	double	Hole diameter of the specimen
dOffset	double	Hole offset

SetCrackedPlateWithNotchProp(double dWidth, double dThick)

Description

Set all Cracked Plate with Notch Specimen properties. **Note:** the notch dimension is not specified since it is determined from the plate width for this case.

Returning value

Return TRUE if function execution was successful; otherwise, FALSE.

Parameters

Name	Type	Description
------	------	-------------

dWidth	double	Width of the specimen
dThick	double	Thickness of the specimen

boolean SetCrkPltWithOffsetCrackProp(double dWidth, double dThick, double dCrackOffset)

Description

Set all Cracked Plate with Offset Crack Specimen properties.

Returning value

Return TRUE if function execution was successful; otherwise, FALSE.

Parameters

Name	Type	Description
dWidth	double	Width of the specimen
dThick	double	Thickness of the specimen
dOffset	double	Crack offset

boolean SetCylinderProp(double dWidth, double dInDia, double dOutDia)¹

Description

Set all Cylinder (Thick Pipe) Specimen properties.

Returning value

Return TRUE if function execution was successful; otherwise, FALSE.

Parameters

Name	Type	Description
dWidth	double	Width of the specimen
dInDia	double	Inner diameter
dOutDia	double	Outer diameter

boolean SetDiscProp(double dThick, double dDia)²

Description

Set all Disk Specimen properties.

Returning value

Return TRUE if function execution was successful; otherwise, FALSE.

Parameters

Name	Type	Description
------	------	-------------

¹ Applies to Internal and External Axial Crack in Thick Pipe in Thick Pipe Weight Function solutions.

² Applies to Through Crack in Disk Weight Function solution.

dThick	double	Thickness of the specimen
dDia	double	Disk diameter

short SetLoad(optional VARIANT dTension, optional VARIANT dBending, optional VARIANT dBearing)

Description

Set the ratio of the tension, bending, or bearing stress ratios to the reference stress for the load case to be modeled. This function is only applicable for models that have multiple load case solutions.

Returning value

A **short** value that specifies the returning result of the function. If it is not equal to 0, the function could not set the load ratios, and the returned result is an error code. All error codes are provided in appendix 1.

Parameters

Name	Type	Description
dTension	VARIANT	ratio of the tension stress to the reference stress
dBending	VARIANT	ratio of the bending stress to the reference stress
dBearing	VARIANT	ratio of the bearing stress to the reference stress

boolean SetLugProp(double dWidth, double dThick, double dDia)

Description

Set all Lug Specimen properties.

Returning value

Return TRUE if function execution was successful; otherwise, FALSE.

Parameters

Name	Type	Description
dWidth	double	Width of the specimen
dThick	double	Thickness of the specimen
dDia	double	Hole diameter of the lug specimen

boolean SetObliqueCrack(double LengthC, double LengthCt)

Description

Set Oblique Crack properties.

Returning value

Return TRUE if function execution was successful; otherwise, FALSE.

Parameters

Name	Type	Description
LengthC	double	Crack length in C(X) direction in AFGROW
LengthCt	double	Short side of the oblique crack length in C(X) direction in AFGROW

boolean SetPipeProp(double dInDia, double dOutDia) ¹

Description

Set all Pipe Specimen properties.

Returning value

Return TRUE if function execution was successful; otherwise, FALSE.

Parameters

Name	Type	Description
dInDia	double	Pipe inner diameter
dOutDia	double	Pipe outer diameter

boolean SetRodProp(double dDia)

Description

Set all Rod Specimen properties.

Returning value

Return TRUE if function execution was successful; otherwise, FALSE.

Parameters

Name	Type	Description
dDia	double	Rod diameter

void SetStartPredictFromCycle(double dCycleNumber)

Description

Sets the cycle number from which AFGROW will start a life prediction. If this cycle number is higher than the number of cycles in the spectrum, AFGROW will halt the prediction and return an error code of 29. This function can set the starting cycle only for the next prediction. Subsequently, AFGROW will start predictions from the beginning of the spectrum.

Returning value

No return value.

¹ Applies to Part-Through and Through Crack in Pipe Standard solution.

Parameters

Name	Type	Description
dCycleNumber	double	Number of the cycle from which the prediction will be started

boolean SetTransitionToOblique()

Description

A Part-Through crack will transition to an Oblique Through Crack. This is valid only for a Single Corner Cracked Hole (*aSingleCornerAtHole*) and Double Corner Cracked Hole (*aDoubleCornerAtHole*).

Returning value

Return **TRUE** if function execution was successful; otherwise, **FALSE**.

boolean SetWOLCTProp(double dWidth, double dThick, double dGrDepth, WOL_CTtype bType)

Description

Set all Common Crack Growth Specimen (CT or WOL) properties.

Returning value

Return **TRUE** if function execution was successful; otherwise, **FALSE**.

Parameters

Name	Type	Description
dWidth	double	Width of the specimen
dThick	double	Thickness of the specimen
dGrDepth	double	Groove Depth of the Common Crack Growth specimen
bType	WOL_CTtype	Type of the Common Crack Growth specimen enumerated in WOL_CTtype (see appendix 2)

boolean SetWFStressDistr(CrackDirection nStressDir, VARIANT aStressDistr)

Description

Set the applied stress distribution for models using the Weight Function Stress Intensity Solution.

Returning value

Return **TRUE** if function execution was successful; otherwise, **FALSE**.

Parameters

Name	Type	Description
nStressDir	CrackDirection	AFGROW crack growth direction enumerated

		in CrackDirection data type (see appendix 2)
aStressDistr	VARIANT	VARIANT 2-Dimensional array filled with stress distribution values for direction defined by nStressDir variable. Each row of the array contains a pair of values (distance from the crack origin and the corresponding stress value). The maximum allowed size of the array is 25x2.

VB example

```
Dim SDarray(2, 1) As Double
SDarray(0, 0) = 0
SDarray(0, 1) = 1
SDarray(1, 0) = 0.11
SDarray(1, 1) = 2
SDarray(2, 0) = 2.45
SDarray(2, 1) = 4
result = mAfgrow.SetWFStressDistr(AYdirection, SDarray)
```

3.3 AFGROW Spectrum

3.2.1 Properties

Name	Type	Description
dPxx	double	The stress value that the structure must be able to maintain at all crack sizes.
SMF	double	Stress Multiplication factor

3.3.2 Functions

boolean ConstAmplitudeSpectrum(double R)

Description

Selects a Constant Amplitude Loading Spectrum in AFGROW.

Returning value

Return TRUE if function execution was successful; otherwise, FALSE.

Parameters

Name	Type	Description
R	double	Stress ratio (Minimum Stress/Maximum Stress)

boolean OpenSpectrumFile(BSTR sFileName)

Description

Opens a new spectrum file in AFGROW.

Returning value

Return TRUE if function execution was successful; otherwise, FALSE.

Parameters

Name	Type	Description
strFileName	BSTR	Full name of the AFGROW spectrum file (*.sp3)

**long GetSpectrumInfo (double* dCycles, double* dLevels, double* dSubspectra,
double* dMaxValue, double* dMinValue)**

(Ignore pointers () in Visual Basic)*

Description

Returns generic information about the spectrum currently open in AFGROW.

Returning value

A **long** value that specifies the result code of the function. If it is not equal to 0, the function finished the prediction, and the returned result is an error code. All error codes are provided in appendix 1.

Parameters

Name	Type	Description
dCycles	double	Number of cycles in the spectrum
dLevels	double	Number of stress levels in the spectrum
dSubspectra	double	Number of sub-spectra in the spectrum
dMaxValue	double	Highest maximum value in the spectrum
dMinValue	double	Lowest minimum value in the spectrum

3.4 AFGROW Materials

3.4.1 Functions

AfgrowMaterials GetActiveMaterial ()

Description

Query the active crack growth rate material data type in AFGROW.

Returning value

Return the active material type in AFGROW. Available AFGROW material data options are enumerated in the AfgrowMaterials data type (see appendix 2).

IFormanMaterialInt* SetFormanMaterial ()

Description

Set the Forman Material data type to be active in AFGROW. The default, or previously set Forman Material data will be used when this material type is activated. To access the properties of this material data type, use IFormanMaterialInt (Forman Material interface in section 10).

Returning value

Return a pointer to the IFormanMaterialInt object (see section 10) if function execution is successful, otherwise, NULL.

IHartertMaterialInt* SetHartertMaterial (optional VARIANT nMaterial)

Description

Set the Harter-T Material data type to be active in AFGROW. The default, or previously set Harter-T data will be used if the optional parameter is omitted. Otherwise, AFGROW will try to select the required material from the AFGROW material database. To access the properties of this material data type, use IHartertMaterialInt (Harter-T material interface in the section 8).

Returning value

Return a pointer to the IHarterTMaterialInt object (see section 8) if function execution is successful, otherwise, NULL.

Parameters

Name	Type	Description
sMaterialName	VARIANT	AFGROW Harter-T material data code (See Appendix 10)

INASGROMaterialInt* SetNASGROMaterial (optional VARIANT sMaterialName)

Description

Set the NASGRO Material data type to be active in AFGROW. The default, or previously set NASGRO material data will be used if the optional parameter is omitted. Otherwise, AFGROW will try to select required material from NASGRO material database. To access the properties of this material data type, use INASGROMaterialInt (see NASGRO material interface in section 12).

Returning value

Return a pointer to the INASGROMaterialInt object (see section 12) if function execution is successful, otherwise, NULL.

Parameters

Name	Type	Description
sMaterialName	VARIANT	NASGRO material code (see appendix 10)

ITabularLookupMaterialInt* SetTabularLookupMaterial ()

Description

Set the Tabular Lookup Material data type to be active in AFGROW. The default, or previously set Tabular Lookup material data will be used when this material type is activated. To access the properties of this material data type, use ITabularLookupMaterialInt (Tabular Lookup material interface in section 10)

Returning value

Return a pointer to the ITabularLookupMaterialInt object (see section 10) if function execution is successful, otherwise, NULL.

IWalkerMaterialInt* SetWalkerMaterial ()

Description

Set the Walker Material data type to be active in AFGROW. The default, or previously set Walker material data will be used when this material type is activated. To access the properties of this material data type, use IWalkerMaterialInt (Walker material interface in section 9).

Returning value

Return a pointer to the IWalkerMaterialInt object (see section 9) if function execution is successful, otherwise, NULL.

3.5 Stress State

3.5.1 Functions

boolean GetStressState(double* dXDirValue, double* dYDirValue)

(ignore pointers in Visual Basic)

Description

Provides access to the current Stress State values in the X (C) and Y (A) growth directions.

Returning value

Returns TRUE if the Automatic Stress State option is selected in AFGROW, otherwise, FALSE.

Parameters

Name	Type	Description
dXDirValue	double*	Stress State value in C direction
dYDirValue	double*	Stress State values in A direction (ignored for through cracks)

boolean SetStressState(boolean bStressStateAuto, VARIANT dXDirValue, VARIANT dYDirValue)

Description

Sets the Stress State calculation mode and the Stress State values in the X and Y directions if applicable.

Returning value

Return TRUE if the Automatic Stress State option has been selected, otherwise, FALSE.

Parameters

Name	Type	Description
bStressStateAuto	boolean	Sets the Stress State calculation mode in

		AFGROW. When it is TRUE, AFGROW calculates the stress state for each crack size during the prediction process; otherwise, AFGROW uses the constant values supplied by the user as the second and third parameters of this function.
dXDirValue	VARIANT	Stress State value in C direction. Used only if the manual Stress State option is selected in AFGROW. If the Auto mode is selected, these values will be ignored.
dYDirValue	VARIANT	Stress State values in A direction, ignored for through cracks. Used only if the Auto Stress State option is selected in AFGROW.

4. Predict Preferences Object Available Parameters and Functions

4.1 Growth Increment Parameters

Properties

Name	Type	Description
bCycleByCycleBetaCalc	boolean	Cycle by cycle beta calculation ¹
bCycleByCycleSpCalc	boolean	Cycle by cycle spectrum calculation ²
dGrowthIncrementPr	double	Growth increment percent

4.2 Output Options

Properties

Name	Type	Description
bOutputToFile	boolean	Output prediction results to a file
bOutputToPlotFile	boolean	Output prediction results to a plot file
sOutputFileName	BSTR	Name of the output file
sOutputPlotFileName	BSTR	Name of the output plot file

4.3 Propagation Limits

4.3.1 Properties

Name	Type	Description
bCrackLengthStop	boolean	Terminate crack propagation at a given crack length in the C direction. Read-only
bCyclesStop	boolean	Terminate crack propagation after a given number of the cycles. Read-only
bKmaxStop	boolean	Terminate crack propagation when K exceeds the maximum K value. Read-only
bNetSectionStop	boolean	Terminate crack propagation when the Net Section Stress exceeds the Yield Stress value. Read-only
bUserKmaxStop	boolean	Terminate crack propagation when K exceeds a given K value. Read-only
dCrackLength	double	Terminate crack propagation at this crack length in the C direction. Read-only
dUserKmax	double	Terminate crack propagation at this K value. Read-only
nCycleCount	long	Terminate crack propagation after this number of cycles. Read-only
nSpectrumToBeRepeated	long	Maximum allowable spectrum repeats (or passes).

4.3.2 Functions

SetPropagationLimits(optional VARIANT bCrackLengthStop, optional VARIANT bCyclesStop, optional VARIANT bKmaxStop, optional VARIANT bUserKmaxStop,

¹ Calculates beta values on a cycle by cycle basis.

² Calculates beta values for a given Vroman crack increment (range) and increment the spectrum in a cycle by cycle basis.

optional VARIANT bNetSectionStop, optional VARIANT dCrackLength, optional VARIANT nCycleCount, optional VARIANT dUserKmax)

Description

Sets Propagation Limits Properties.

Parameters

Name	Type	Description
bCrackLengthStop	VARIANT	Terminate crack propagation at given crack length in the C direction.
bCyclesStop	VARIANT	Terminate crack propagation after a given number of the cycles.
bKmaxStop	VARIANT	Terminate crack propagation when K exceeds the maximum K value.
bNetSectionStop	VARIANT	Terminate crack propagation when the Net Section Stress exceeds the Yield Stress value.
bUserKmaxStop	VARIANT	Terminate crack propagation when K exceeds a given K value.
dCrackLength	VARIANT	Terminate crack propagation at this crack length in the C direction.
dUserKmax	VARIANT	Terminate crack propagation after this number of cycles.
nCycleCount	VARIANT	Terminate crack propagation after this number of cycles.

4.4 Part Through to Through Crack Transition Options

Properties

Name	Type	Description
nTransitionAt	CrackTransConditions	Sets criteria for transition from part through to through crack. Available transition criteria are enumerated in CrackTransConditions data type (see appendix 2).

4.5 Output Intervals

4.5.1 Properties

Name	Type	Description
bLifeInHours	boolean	Specifies the option to display the lifetime in hours (Read-only , use SetLifeInHours function to set this option).
dCrackGrowthIncrement	double	Specifies the change in crack length between two sequential data outputs. Used only when <i>nOutputIntervalType</i> set to <i>CrackGrowthIncrement</i> . Read-only
dHoursPerPass	double	Parameter that specifies the number of hours per pass through the spectrum. Used only when the <i>LifeInHours</i> option is selected (Read-only , use SetLifeInHours function to set this option).
nSpectrumCyclicIncrement	long	Specifies the number of cycles between two sequential data outputs. Used only when <i>nOutputIntervalType</i> set to <i>SpectrumCyclicIncrement</i> . Read-only
nOutputIntervalType	OutputIntervalType	Parameter that determines the type of output interval. All possible options (crack length, number of cycles, or each spectrum stress level) are enumerated in the <i>OutputIntervalType</i> data type (see appendix 2). Read-only

4.5.2 Functions

SetLifeInHours(boolean bLifeInHours, [in, optional] VARIANT dHoursPerPass);

Description

Sets Properties for Displaying Lifetime in Hours.

Parameters

Name	Type	Description
bLifeInHours	boolean	Turns the Display Lifetime in Hours option ON or OFF in AFGROW.
dHoursPerPass	VARIANT	The number of hours per pass through the spectrum. This value is used to calculate the lifetime in hours. This parameter is optional. It should be ignored if <i>bLifeInHours</i> parameter is FALSE.

SetOutputIntervals(OutputIntervalType nIntervalType, [in, optional] VARIANT aIntervalValue)

Description

Sets Output Interval Properties.

Parameters

Name	Type	Description
nIntervalType	OutputIntervalType	Parameter that determines the type of output interval. All possible options (crack length, number of cycles, or each spectrum stress level) are enumerated in the <i>OutputIntervalType</i> data type (see appendix 2).
aIntervalValue	VARIANT	The output interval to be used for the given type of the output interval (crack length or cycles). This parameter is optional. If it is omitted, AFGROW will use the most recently set value. That value will be ignored if the option: <i>EachSpectrumStressLevel</i> is selected. Note: An error will result if the interval is not appropriate for the given output interval type.

5. Beta Correction Object Available Parameters and Functions

Properties

Name	Type	Description
aBetaCorrectionData	VARIANT	Beta correction data. It is a 2-Dimensional array filled with Normalized Stress or Beta Correction distribution values. (Data type is defined by nTypeOfData variable). Maximum allowed size of the array is 25x2 or 25x3. Each row of the array contains a distance from the crack origin and a corresponding beta correction value. An array with two columns contains only data for the X direction and an array with three columns contains data for the X and Y direction. If a three-column array is used for a through crack, the Y direction will be ignored. The first distance value should be greater than 0.
bUseBetaCorrection	boolean	Turns the beta correction option ON or OFF in AFGROW (The Beta correction option can not be used for Weight Function Stress Intensity Solution Models).
nTypeOfData	TypeOfBetaCorData	Sets the type of the beta correction data (Normalized Stresses or Beta Correction Factors). Available data types are enumerated in TypeOfBetaCorData (see appendix 2).

VB sample

```

Dim SDarray(3, 1) As Double
Dim SBeta As Variant
On Error GoTo error
BetaCor.bUseBetaCorrection = True

SDarray(0, 0) = 0.1
SDarray(0, 1) = 1
SDarray(1, 0) = 0.11
SDarray(1, 1) = 2
SDarray(2, 0) = 2.45
SDarray(2, 1) = 4
SDarray(3, 0) = 3.45
SDarray(3, 1) = 4.6

SBeta = SDarray
BetaCor.aBetaCorrectionData = SBeta
BetaCor.nTypeOfData = NormalizedStress
SBeta = BetaCor.aBetaCorrectionData
BetaCor.nTypeOfData = BetaCorrectionFactors

```

BetaCor.aBetaCorrectionData = SBeta

6. Residual Stress Object Available Parameters and Functions

Properties

Name	Type	Description
bApplyResidualStresses	boolean	Turns the residual stress option ON or OFF in AFGROW.
aResidualStressesData	VARIANT	<p>Residual stresses data. It is a 2-Dimensional array filled with Stress or Residual K distribution values (Data type is defined by nTypeOfData variable). Maximum allowed size of the array is 25x2 or 25x3.</p> <p>Each row of the array contains a distance from the crack origin and corresponding residual stress values. An array with two columns contains only data for the X direction and an array with three columns contains data for the X and Y direction. If a three-column array is used for a through crack, the Y direction will be ignored.</p>
nTypeOfSIFTTableGen	TypeOfResSIFTTableGeneration	<p>Sets the method for residual stress intensity (SIF) table calculation inside AFGROW. Available data types are enumerated in TypeOfResSIFTTableGeneration (see appendix 2). The Weight Function method of residual SIF calculation is possible only if a weight function solution is available for the geometry being analyzed. Currently this option can be used for all AFGROW Weight Function models plus for 1010, 1070, 2010, 2040, and 2050 Standard Solution models. The Gaussian integration method can be used for all models.</p>
nTypeOfData	TypeOfResStressData	<p>Sets the type of the residual stresses data (Stresses or Residual K). Available data types are enumerated in TypeOfResStressData (see appendix 2).</p>

VB sample

```
Dim SDarray(5, 2) As Double
Dim SResid As Variant
On Error GoTo error
ResStress.bApplyResidualStresses = True
ResStress.nTypeOfData = Stress
SDarray(0, 0) = 0.1
SDarray(0, 1) = 1
SDarray(0, 2) = 1.1
SDarray(1, 0) = 0.11
SDarray(1, 1) = 2
SDarray(1, 2) = 2.1
SDarray(2, 0) = 2.45
SDarray(2, 1) = 4
SDarray(2, 2) = 4.1
SDarray(3, 0) = 3.12
SDarray(3, 1) = 5.1
SDarray(3, 2) = 5.1
SDarray(4, 0) = 3.5
SDarray(4, 1) = 2.1
SDarray(4, 2) = 1.1
SDarray(5, 0) = 3.9
SDarray(5, 1) = 2.2
SDarray(5, 2) = 1.2
SResid = SDarray
ResStress.aResidualStressesData = SDarray
SResid = ResStress.aResidualStressesData
ResStress.nTypeOfData = ResidualK
ResStress.aResidualStressesData = SResid
```

7. User Defined Beta Object Available Parameters and Functions

7.1 Properties

Name	Type	Description
aThroughCrackData	VARIANT	<p>User Defined Beta data used only for through cracks. It is a 2-Dimensional array containing user-defined Beta values. The maximum allowed size of the array is 25x2.</p> <p>Each row of the array contains a distance from the crack origin and a corresponding user-defined beta value.</p>

7.2 Functions


long GetFourPointsInt(double* dA12, double* dA34, double* dC1, double* dC2, double* dC3, double* dC4, double* dBetaA1, double* dBetaA2, double* dBetaA3, double* dBetaA4, double* dBetaC1, double* dBetaC2, double* dBetaC3, double* dBetaC4)

(Ignore pointers () in Visual Basic)*

Description

Get information for the Four Point User-Defined Beta Interpolation Method.

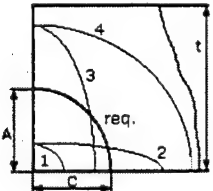
Four point Interpolation

 Enter Beta values for four reference cracks that encompass the expected range of crack growth. Cracks no. 1 and 2 must have an equal 'A' value that is less than or equal to the initial crack depth, 'A'.

Cracks no. 3 and 4 must have an equal 'A' value that is greater than or equal to 95% of the thickness.

Also, cracks no. 1 and 3 must have 'C' values less than or equal to the initial crack length, 'C', and cracks no. 2 and 4 must have 'C' values greater than or equal to the maximum crack length, expected before transition to a through-crack.

During fatigue life calculations, if the current crack falls outside this envelope, it will be assigned Beta values equal to those of the nearest reference crack.



no.	'A'	'C'	Beta 'A'	Beta 'C'
1	0.05	0.07	1	1
2		0.475	1	1
3	0.2375	0.07	1	1
4		0.475	1	1

< Back
Next >
Cancel
Help

Figure 4: Four Point Interpolation Page of the User-Defined Beta Wizard

Returning value

A **long** value that specifies the returning result of the function. If it is not equal to 0, there was an error in obtaining the requested information. All error codes are provided in appendix 1.

Parameters

Name	Type	Description
dA12	double*	The a-length for the first and second reference crack
dA34	double*	The a-length for the third and fourth reference crack
dC1	double*	The c-length for the first reference crack
dC2	double*	The c-length for the second reference crack
dC3	double*	The c-length for the third reference crack
dC4	double*	The c-length for the fourth reference crack
dBetaA1	double*	Beta (a-dimension) for the first reference crack
dBetaA2	double*	Beta (a-dimension) for the second reference crack
dBetaA3	double*	Beta (a-dimension) for the third reference crack
dBetaA4	double*	Beta (a-dimension) for the fourth reference crack
dBetaC1	double*	Beta (c-dimension) for the first reference crack
dBetaC2	double*	Beta (c-dimension) for the second reference crack
dBetaC3	double*	Beta (c-dimension) for the third reference crack
dBetaC4	double*	Beta (c-dimension) for the fourth reference crack

long GetLinealnt(VARIANT* aALengthArray, VARIANT* aCLengthArray, VARIANT* aABetasArray, VARIANT* aCBetasArray)

(Ignore pointers () in Visual Basic)*

Description

Get information for the User-Defined Beta Linear Interpolation Method.

Returning value

A **long** value that specifies the returning result of the function. If it is not equal to 0, there was an error in obtaining the requested information. All error codes are provided in appendix 1.

Parameters

Name	Type	Description
aLengthArray	VARIANT*	Range of A Length values for interpolation (No more than 100 and there should be an equal number of C Length values).
aCLengthArray	VARIANT *	Range of C Length values for interpolation (No more than 100 and there should be an equal number of A Length values).
aBetasArray	VARIANT *	2-D data matrix containing A Beta values for the range specified by aLengthArray and aCLengthArray (No more than 100x100 size).
aCBetasArray	VARIANT *	2-D data matrix containing C Beta values for the range specified by aLengthArray and aCLengthArray (No more than 100x100 size).

TypeOfUserDefBetaPartThroughCrackInt GetPartThroughIntType ()

Description

Determine the type of the User Defined Beta interpolation method being used for a part-through crack.

Returning value

Type of the User-Defined Beta interpolation method being used for part-through cracks. These values are enumerated in TypeOfUserDefBetaPartThroughCrackInt (See appendix 2).

boolean IsSet()

Description

Determine whether the User-Defined Beta information is set in AFGROW (If this information is required and has not been set, AFGROW will refuse to execute the crack growth analysis module).

Note: When setting User-Defined Beta information for the Part-Through and Through cracks it is acceptable to set only the User Defined Beta data for a through crack (*aThroughCrackData*). If the part-through information is not set, AFGROW will use the default Four Point Interpolation data. This means that users may set information using only the *aThroughCrackData* property and obtain a positive result (TRUE) from this function.

Returning value

Returns TRUE if the User Defined Beta information has been set in AFGROW; otherwise, FALSE.

long SetFourPointsInt(double dA12, double dA34, double dC1, double dC2, double dC3, double dC4, double dBetaA1, double dBetaA2, double dBetaA3, double dBetaA4, double dBetaC1, double dBetaC2, double dBetaC3, double dBetaC4)

Description

Set information for the Four Point User-Defined Beta Interpolation method in AFGROW (see figure 4).

Returning value

A **long** value that specifies the returning result of the function. If it is not equal to 0, there was an error setting the required information. All error codes are provided in appendix 1.

Parameters

Name	Type	Description
dA12	double	The a-length for the first and second reference crack
dA34	double	The a-length for the third and fourth reference crack
dC1	double	The c-length for the first reference crack
dC2	double	The c-length for the second reference crack
dC3	double	The c-length for the third reference crack
dC4	double	The c-length for the fourth reference crack
dBetaA1	double	Beta (a-dimension) for the first reference crack
dBetaA2	double	Beta (a-dimension) for the second reference crack
dBetaA3	double	Beta (a-dimension) for the third reference crack
dBetaA4	double	Beta (a-dimension) for the fourth reference crack
dBetaC1	double	Beta (c-dimension) for the first reference crack
dBetaC2	double	Beta (c-dimension) for the second reference crack
dBetaC3	double	Beta (c-dimension) for the third reference crack
dBetaC4	double	Beta (c-dimension) for the fourth reference crack

long SetLinealnt(VARIANT aALengthArray, VARIANT aCLengthArray, VARIANT aABetasArray, VARIANT aCBetasArray)

Description

Set information for the User-Defined Beta Linear Interpolation method in AFGROW.

Returning value

A **long** value that specifies the returning result of the function. If it is not equal to 0, there was an error setting the required information. All error codes are provided in appendix 1.

Parameters

Name	Type	Description
aLengthArray	VARIANT	Range of A Length values for interpolation (No more than 100 and should be equal number of C Length values).
aCLengthArray	VARIANT	Range of C Length values for interpolation (No more than 100 and should be equal number of A Length values).
aBetasArray	VARIANT	2-D data matrix with A Beta values for the range specified by aLengthArray and aCLengthArray (No more than 100x100 size).
aCBetasArray	VARIANT	2-D data matrix with A Beta values for the range specified by aLengthArray and aCLengthArray (No more than 100x100 size).

VB sample

```
Private Sub bUserBeta_Click()  
Dim selected As Boolean  
Dim IntType As Afgrow.TypeOfUserDefBetaPartThroughCrackInt  
Dim result As Integer  
Dim a12 As Double  
Dim a34 As Double  
Dim c1 As Double  
Dim c2 As Double  
Dim c3 As Double  
Dim c4 As Double  
Dim ba1 As Double  
Dim ba2 As Double  
Dim ba3 As Double  
Dim ba4 As Double  
Dim bc1 As Double  
Dim bc2 As Double  
Dim bc3 As Double  
Dim bc4 As Double  
Dim SDarray(3, 1) As Double  
Dim SBeta As Variant  
On Error GoTo error  
  
'Check set data  
SDarray(0, 0) = 0.1  
SDarray(0, 1) = 1  
  
SDarray(1, 0) = 0.11  
SDarray(1, 1) = 2  
  
SDarray(2, 0) = 2.45  
SDarray(2, 1) = 4  
  
SDarray(3, 0) = 3.45  
SDarray(3, 1) = 4.6  
  
SBeta = SDarray  
  
    selected = UserDefBeta.IsSet  
    If (selected = False) Then  
        MsgBox ("selected = FALSE")
```

```

        UserDefBeta.aThroughCrackData = SBeta
    End If
    IntType = UserDefBeta.GetPartThroughIntType

    MsgBox ("DataType = " & CStr(IntType))

'---Checking Part through Crack
    result = UserDefBeta.GetFourPointsInt(a12, a34, c1, c2, c3, c4,
    ba1, ba2, ba3, ba4, bc1, bc2, bc3, bc4)
    result = UserDefBeta.SetFourPointsInt(0.05, 0.2375, 0.07, 0.475,
    0.07, 0.475, 1#, 1.2, 1#, 1.2, 1, 1.2, 1#, 1.2)

'---Checking Linear Interpolation
    Dim a(1) As Double
    Dim c(1) As Double
    Dim aBeta(1, 1) As Double
    Dim cBeta(1, 1) As Double

    'Check for set data
    a(0) = 0.05
    a(1) = 0.235

    c(0) = 0.07
    c(1) = 0.271

    aBeta(0, 0) = 0.842
    aBeta(0, 1) = 0.327

    aBeta(1, 0) = 1.159
    aBeta(1, 1) = 0.973

    cBeta(0, 0) = 0.613
    cBeta(0, 1) = 1.083

    cBeta(1, 0) = 0.227
    cBeta(1, 1) = 0.979

    result = UserDefBeta.SetLineaInt(a, c, aBeta, cBeta)

    Dim a1 As Variant
    Dim cc1 As Variant
    Dim baal As Variant
    Dim bccl As Variant
    result = UserDefBeta.GetLineaInt(a1, cc1, baal, bccl)

    Dim Lower As Integer
    Lower = LBound(a1, 1)
    Lower = UBound(cc1, 1)
    result = UserDefBeta.SetLineaInt(a1, cc1, baal, bccl)
Exit Sub
error: If Err.Number <> 0 Then MsgBox "Message from " + Err.Source + "
:" + Err.Description
End Sub

```

8. Harter-T Material Object Available Parameters and Functions

Functions

boolean IsActive ()

Description

Determine whether this material object is active in AFGROW.

Returning value

TRUE if the Harter-T Material is active in AFGROW and FALSE if not.

boolean SetActive(VARIANT nMaterial)

Description

Set the Harter-T Material data type to be active in AFGROW. The program will use the default AFGROW data or previously used Harter-T material if the optional parameter is omitted. If the parameter is included, the specified material data will be selected from the AFGROW material database.

Returning value

Return TRUE if function execution was successful; otherwise, FALSE.

Parameters

Name	Type	Description
nMaterial	VARIANT	AFGROW Harter-T material data code (see appendix 10)

9. Walker Material Object Available Parameters and Functions

9.1 Properties

Name	Type	Description
aCurveSegments	VARIANT	2-D data matrix of Walker Equation data. Each row in this matrix corresponds to one segment of dadN curve and may contain no more than 5 segments. Each segment entry consists of: C : Value of da/dN when R=0 and Delta K=1 (da/dN intercept) n : Paris Exponent (da/dN slope) m : Walker exponent
dKlc	double	Plane strain fracture toughness
dKc	double	Plane stress fracture toughness
dPoissonsRatio	double	Poisson's ratio
dRhi	double	Upper limit on R shift
dRlo	double	Lower limit on R shift
dThermalEx	double	Coefficient of thermal expansion
dThreshold	double	Delta K Threshold Value at R=0
sMaterialName	BSTR	Material name
dYield	double	Yield strength
dYoungModulus	double	Young's modulus

9.2 Functions

boolean IsActive ()

Description

Determine whether this material object is active in AFGROW.

Returning value

TRUE if the Walker Material data type is active in AFGROW and FALSE if not.

boolean SetActive()

Description

Set the Walker Material data type to be active in AFGROW.

Returning value

Return TRUE if function execution was successful; otherwise, FALSE.

10. Forman Material Object Available Parameters and Functions

10.1 Properties

Name	Type	Description
aCurveSegments	VARIANT	2-D data matrix of Forman Equation data for the first (main) data fit. Each row in this matrix corresponds to one segment of dadN curve and it can contain data for no more than 3 segments. Each segment entry consists of: C: Value of da/dN when R=0 and Delta K=1 (da/dN intercept) n: Paris Exponent (in this case, limit in da/dN slope as DK approaches 0.0) Kcut: Value of Delta K (at R=0) defining the highest Delta K allowed for the given segment (upper segment boundary) Note: The Kcut value for the last defined segment is assumed to be equal to the plane stress fracture toughness of the metal being analyzed.
dKlc	double	Plane strain fracture toughness
dPoissonsRatio	double	Poisson's ratio
dRhi	double	Upper limit on R shift
dRlo	double	Lower limit on R shift
dThermalEx	double	Coefficient of thermal expansion
dThreshold	double	Delta K Threshold Value at R=0
sMaterialName	BSTR	Material name
dYield	double	Yield strength
dYoungModulus	double	Young's modulus

10.2 Functions

boolean IsActive ()

Description

Determine whether this material object is active in AFGROW.

Returning value

TRUE if the Forman Material is active in AFGROW and FALSE if not.

boolean GetMapR([out] double* dRloBorder, [out] double* dRhiBorder,[out] double* dRValue)

(Ignore pointers () in Visual Basic)*

Description

AFGROW allows users to map the Forman fit for a range of R values to a given R. This function queries parameters for this R mapping to determine whether the mapping option is active.

Returning value

Return TRUE if the Map Forman R option in AFGROW is active; otherwise, FALSE.

Parameters

Name	Type	Description
dRhiBorder	double*	Upper limit of the R range
dRloBorder	double*	Lower limit of the R range
dRValue	double*	R value that R range will be mapped to

boolean GetUserRRange(double* dRValue, VARIANT* aCurveSegments)

(Ignore pointers () in Visual Basic)*

Description

AFGROW allows users to define up to two fits (three segments each) as a function of stress ratio (R). This function queries parameters for of the second stress ratio fit mapping to determine whether the second fit option is being used.

Returning value

Return TRUE if AFGROW uses two fit curve is active; otherwise, FALSE.

Parameters

Name	Type	Description
dRValue	double*	Rcut (AFGROW uses second fit when R is greater than Rcut)
aCurveSegments	VARIANT*	2-D data matrix of Forman Equation data for the second data fit. Each row in this matrix corresponds to one segment of dadN curve and it can contain data for no more than three segments. Each segment entry consists of: C: Value of da/dN when $R=0$ and $\Delta K=1$ (da/dN intercept) n: Paris Exponent (in this case, limit in da/dN slope as ΔK approaches 0.0) Kcut: Value of ΔK (at $R=0$) defining the highest ΔK allowed for the given segment (upper segment boundary) Note: The Kcut for the last defined segment is assumed to be equal to the plane stress fracture toughness of the metal being analyzed.

boolean SetActive()

Description

Sets the Forman Material data type to be active in AFGROW.

Returning value

Return TRUE if function execution was successful; otherwise, FALSE.

long SetMapR(boolean bMapR, optional VARIANT dRloBorder, optional VARIANT dRhiBorder, optional VARIANT dRValue)

Description

AFGROW allows users to map the Forman fit for a range of R values to a given R. This function turns mapping ON/OFF and set parameters for the R mapping.

Returning value

A **long** value that specifies the returning result of the function. If it is not equal to 0, the function was unable to set the desired mapping parameters. All error codes are provided in appendix 1.

Parameters

Name	Type	Description
bMapR	boolean	Set mapping of the R in AFGROW Forman Material to be active (TRUE) or not (FALSE).
dRhiBorder	double	Upper limit of the R range. It is optional value, used only when mapping set to be TRUE.
dRloBorder	double	Lower limit of the R range. It is optional value, used only when mapping set to be TRUE.
dRValue	double	R value that the R range will be mapped to. It is an optional value, used only when mapping set to be TRUE.

long SetUseRRange(boolean bUserRRange, VARIANT dRValue, VARIANT aCurveSegments)

Description

AFGROW allows users to define up to two fits (three segments each) as a function of stress ratio (R). This function sets parameters for of the second stress ratio fit.

Returning value

A **long** value that specifies the returning result of the function. If it is not equal to 0, the function was unable to set the parameters for the second R fit. All error codes are provided in appendix 1.

Parameters

Name	Type	Description
bUseRRange	boolean	Set the R cut option for the AFGROW Forman material data to be active (TRUE) or not (FALSE).
dRValue	double	Rcut (AFGROW uses a second fit when R is greater than this value). It is an optional value, used only when <i>UseRRange</i> is set to be TRUE.
aCurveSegments	VARIANT	2-D data matrix of Forman Equation data for the second data fit. It is an optional value, used only when <i>UseRRange</i> set to be TRUE. Each row in this matrix corresponds to one segment of the dadN curve and it can contain data for no more than three segments. Each segment entry consists of: C: Value of da/dN when R=0 and Delta K=1 (da/dN intercept) n: Paris Exponent (in this case, limit in da/dN slope as DK approaches 0.0) Kcut: Value of Delta K (at R=0) defining the highest Delta K allowed for the given segment (upper segment boundary) Note: The Kcut value for the last defined segment is assumed to be equal to the plane stress fracture toughness of the metal being analyzed.

11. TabularLookup Material Object Available Parameters and Functions

11.1 Properties

Name	Type	Description
aDataMatrix	VARIANT	2-D data matrix of Tabular Lookup data. The first row in this matrix contains the range of R values beginning from 1 (not 0). The first column contains the range of dadN values beginning from 1 (not 0). The ΔK (K_{max} if $R < 0$) data are entered in columns below their corresponding R value. Note: The data value in position (0,0) is never used. The Maximum size of the matrix is 26x11 corresponding to a 25x10 ΔK matrix.
ddadNhi	double	Upper limit on da/dN
ddadNlo	double	Lower limit on da/dN
dKc	double	Plane stress fracture toughness
dKlc	double	Plane strain fracture toughness
dPoissonsRatio	double	Poisson's ratio
dRhi	double	Upper limit on R shift
dRlo	double	Lower limit on R shift
dThermalEx	double	Coefficient of thermal expansion
dThreshold	double	Delta K Threshold Value at R=0
sMaterialName	BSTR	Material name
dYield	double	Yield strength
dYoungModulus	double	Young's modulus

11.2 Functions

boolean IsActive ()

Description

Determine whether this material object is active in AFGROW.

Returning value

TRUE if Tabular Lookup Material was active in AFGROW; otherwise, FALSE.

boolean SetActive()

Description

Set the Tabular Material data type to be active in AFGROW.

Returning value

Return TRUE if function execution was successful; otherwise, FALSE.

boolean SetData(VARIANT aDataMatrix, double dRlo, double dRhi, double ddadNlo, double ddadNhi, double dThreshold)

Description

Set the Tabular Lookup Material data in AFGROW.

Returning value

Return TRUE if the function execution was successful; otherwise, FALSE.

Parameters

Name	Type	Description
aDataMatrix	VARIANT	2-D data matrix of Tabular Lookup data. The first row in this matrix contains the range of R values beginning from 1 (not 0). The first column contains the range of dadN values beginning from 1 (not 0). The ΔK (K_{max} if $R < 0$) data are entered in columns below their corresponding R value. Note: The data value in position (0,0) is never used. The Maximum size of the matrix is 26x11 corresponding to a 25x10 ΔK matrix.
ddadNhi	double	Upper limit on da/dN
ddadNlo	double	Lower limit on da/dN
dRhi	double	Upper limit on R shift
dRlo	double	Lower limit on R shift
dThreshold	double	Delta K Threshold Value at R=0

12. NASGRO Material Object Available Parameters and Functions

Functions

boolean IsActive ()

Description

Determine whether this material object is active in AFGROW.

Returning value

TRUE if NASGRO Material is active in AFGROW; otherwise, FALSE.

boolean SetActive(VARIANT sMaterialName)

Description

Set the NASGRO Material data type to be active in AFGROW. The default, or previously set NASGRO material data will be used if the optional parameter is omitted. Otherwise, AFGROW will try to select required material from NASGRO material database.

Returning value

Return TRUE if the function execution was successful; otherwise, FALSE.

Parameters

Name	Type	Description
sMaterialName	VARIANT	AFGROW NASGRO material data code (see appendix 10)

13. Crack Initiation Object Available Parameters and Functions

13.1 Properties

Name	Type	Description
aCyclicUserDefData	VARIANT	2-D data matrix of the Cyclic Stress-Strain data. Maximum allowed size of the array is 25x2. Each row of the array contains a strain and a corresponding stress value.
aStrainLifeUserDefinedData	VARIANT	2-D data matrix of the Strain-Life data. Maximum allowed size of the array is 25x2. Each row of the array contains a strain and a corresponding life value. The first data pair must be the strain value for failure at one reversal (life is given in reversals = cycles/2).
bUseCrackInitiation	boolean	Turn ON or OFF initiation analysis
dCyclicStrainExp	double	Cyclic Strain Hardening Exponent
dCyclicStrengthCoef	double	Cyclic Strength Coefficient
dFatigueDuctCoef	double	Fatigue Ductility Coefficient
dFatigueDuctExp	double	Fatigue Ductility Exponent
dFatigueNotchConst	double	Fatigue Notch Constant
dFatigueStrengthCoef	double	Fatigue Strength Coefficient
dFatigueStrengthEsp	double	Fatigue Strength Exponent
dNotchRadius	double	Notch Radius (Read-only for all cracks at hole specimens)
dStressConcFactor	double	Stress concentration factor $\sigma_{local}/\sigma_{ref}$
sMaterialName	BSTR	Material name

13.2 Functions

DataType GetCyclicDataType ()

Description

Determine the type of the Cyclic Stress-Strain Initiation data.

Returning value

Gets the type of the Cyclic Stress-Strain Initiation data (user defined or not). Available data types are enumerated in DataType (see appendix 2).

DataType Get GetStrainLifeDataType Type ()

Description

Determine the type of the Strain-Life Initiation data.

Returning value

Gets type of the Cyclic Strain-Life Initiation data (user defined or not). Available data types are enumerated in DataType (see appendix 2).

DataType SetCyclicDataType (DataType nDataType, [in, optional] VARIANT aData)

Description

Set the type of the Cyclic Stress-Strain Initiation data (see appendix 2) and, if applicable, any user-defined data.

Returning value

Return the previous type of the data.

Parameters

Name	Type	Description
nDataType	DataType	Cyclic Stress-Strain Data Type. Available data types enumerated in DataType (see appendix 2).
aData	VARIANT	2-D data matrix of the Cyclic Stress-Strain data. Maximum allowed size of the array is 25x2. Each row of the array contains a strain and a corresponding stress value. Optional parameter used only when user sets User-Defined data.

DataType SetStrainLifeDataType (DataType nDataType, [in, optional] VARIANT aData)

Description

Set a type of the Strain-Life Initiation data (see appendix 2) and if applicable, any user-defined data.

Returning value

Return previous type of the data.

Parameters

Name	Type	Description
nDataType	DataType	Strain-Life Data Type. Available data types enumerated in DataType (see appendix 2).
aData	VARIANT	2-D data matrix of the Strain-Life data. Maximum allowed size of the array is 25x2. Each row of the array contains a strain and a corresponding stress value. The first pair of the strain-life values must be the strain value for failure at one reversal (life is given in reversals = cycles/2). Optional parameter used only when user sets User-Defined data.

14. Retardation Object Available Parameters and Functions

14.1 Properties

Name	Type	Description
------	------	-------------

dWheelerParam	double	Wheeler retardation model shaping exponent. Read only
dWillenborgParam	double	Willenborg retardation model Shut-off Overload Ratio. Read only

14.2 Functions

AfgrowRetardationModels GetActiveRetardationModel()

Description

Gets the Active (Selected) retardation model in AFGROW.

Returning value

Returns the Active (Selected) retardation model in AFGROW. AFGROW retardation models are enumerated in AfgrowRetardationModels data type (see appendix 2).

long GetClosureModelParam(double* dParameter, boolean* bOpenLoadRatioAuto, double* dOpenLoadRatio)

(Ignore pointers in Visual Basic)

Description

Gets the Active Closure retardation model parameters.

Returning value

Reserved parameter, always equal to 0.

Parameters

Name	Type	Description
dParameter	double*	Closure retardation parameter (opening load ratio)
bOpenLoadRatioAuto	boolean*	Used to specify the initial crack opening level for Closure Retardation model. If This parameter is TRUE, the program uses the first cycle of the input spectrum to determine the Initial Opening Load Ratio parameter. If it is FALSE, AFGROW uses dOpenLoadRatio parameter as the initial crack opening level.
dOpenLoadRatio	double*	Initial crack opening level. Used only when bOpenLoadRatioAuto is FALSE.

AfgrowRetardationModels NoRetardation()

Description

Sets no retardation in AFGROW.

Returning value

The previously selected retardation model in AFGROW.

AfgrowRetardationModels SetClosureModel(VARIANT dParameter, VARIANT bOpenLoadRatioAuto, VARIANT dOpenLoadRatio)

Description

Sets the Closure retardation model in AFGROW.

Returning value

The previously selected retardation model in AFGROW.

Parameters

Name	Type	Description
dParameter	VARIANT	Closure retardation parameter (opening load ratio)
bOpenLoadRatioAuto	VARIANT	Used to specify initial crack opening level for Closure Retardation model. If This parameter is TRUE, program uses first cycle of the input spectrum to determine the Initial Opening Load Ratio parameter. If it is FALSE, AFGROW expects user to set <i>dOpenLoadRatio</i> parameter in this function and will use it as the initial crack opening level.
dOpenLoadRatio	VARIANT	Initial crack opening level. Used only when <i>bOpenLoadRatioAuto</i> is FALSE.

AfgrowRetardationModels SetWheelerModel(VARIANT dParameter)

Description

Sets the Wheeler retardation model in AFGROW.

Returning value

The previously selected retardation model in AFGROW.

Parameters

Name	Type	Description
dParameter	VARIANT	Wheeler retardation parameter (Wheeler shaping exponent - m)

AfgrowRetardationModels SetWillenborgModel(VARIANT dParameter)

Description

Sets the Willenborg retardation model in AFGROW.

Returning value

The previously selected retardation model in AFGROW.

Parameters

Name	Type	Description
dParameter	VARIANT	Willenborg retardation parameter (shut off overload ratio)

15. Output Object Parameters

15.1 Properties

Name	Type	Description
dBeta	double	Beta
dCrackLength	double	Crack length
ddadN	double	Crack Growth Rate
dDeltaK	double	Delta K
dKres	double	Residual K
dPSX	double	Stress State
dREffective	double	Effective Stress Ratio used to calculate da/dN (after any retardation model is applied).
dRK	double	K min/K max Ratio
nDimension	CrackDirection	AFGROW crack growth direction enumerated in CrackDirection data type (see appendix 2).

15.2 VB sample

```
Private Sub mAfgrow_PredictInfo(ByVal Model As Afgrow.AfgrowModels,
ByVal dStress As Double, ByVal dRStress As Double, ByVal dCycles As
Double, ByVal dPass As Long, ByVal iOutputInfo1 As Object, ByVal
iOutputInfo2 As Object, ByVal iOutputInfo3 As Object, ByVal
iOutputInfo4 As Object)

Dim temp As Double
Dim oInfo As Afgrow.OutputInfoInt

Set oInfo = iOutputInfo1
pBar.Value = oInfo.dCrackLength
If nCount < 50 Then
    nCount = nCount + 1
    CrLength(nCount) = CDBl(Format(oInfo.dCrackLength, "00.000000"))
    Cycles(nCount) = dCycles
    Beta(nCount) = oInfo.dBeta
End If
Set iOutputInfo = Nothing
Set oInfo = Nothing
End Sub
```

Appendix 1: Return Codes

Application Object - Predict Function Return Codes

Number	Error Description
-1	Unknown error
0	Normal completion of the Predict routine
1	After one pass of the spectrum, growth was less than 1.0e-13. Program halted
2	Maximum Number of passes exceeded
3	Crack length exceeded stop value
4	Cycle count exceeded stop value
5	Fracture based on User-Defined 'Kmax' Criteria
10	No Spectrum file specified... Cannot Predict!
11	Beta table has zero length... Cannot Predict!
12	Repair patch applied and initiation on... Cannot Predict!
13	Initiation is not allowed with this model... Cannot Predict!
14	NASGRO equation Rhi value is to large
15	Beta correction is not allowed with this model. Can not Predict
16	Residual Stresses are not allowed with this model. Can not Predict
20	Wrong geometry for an oblique crack
21	The initial crack length in the thickness direction is greater than the input thickness. Check data.
22	Error in spectrum file(s)
23	Unable to open plotfile
24	Error in initiation. Predict stopped
25	The number of cycles to initiation is greater than 2.e+9. Predict stopped
26	Not enough memory to allocate spectrum
27	Only a BLOCKED spectrum may have more than one (1) cycle per stress level
28	Program termination by user
29	The total number of spectrum cycles was less than the user-specified cycle number to start the prediction
30	KIC greater than KC

Application Object - SetLoad Function Return Codes

Number	Error Description
0	Normal completion of the SetLoad routine
1	Invalid tension ratio value
2	Invalid bending ratio value
3	Invalid bearing ratio value
4	This model does not have a multiple loads case solution
5	Filled unloaded hole is selected. Cannot set load
6	This model does not have a tension load
7	This model does not have a bending load
8	This model does not have a bearing load

User-Defined Beta Object - Interpolation Function Return Codes

Number	Error Description
0	Normal completion of the routine.
1	Wrong AFGROW model.
2	Four points interpolation is not selected in AFGROW.
3	Crack numbers 1 and 2 must have 'A' value that is less than or equal to the initial crack depth.
4	Crack numbers 3 and 4 must have 'A' value that is greater or equal to 95% of the thickness.
5	Crack number 1 must have a 'C' value that is less than or equal to the initial crack length.
6	Crack number 3 must have a 'C' value that is less than or equal to the initial crack length.
7	Crack number 3 must have a 'C' value that is less than crack number 4 'C' value.
8	Crack number 1 must have a 'C' value that is less than crack number 2 'C' value.
9	Crack length value is wrong (usually less or equal to 0)
10	Cannot allocate memory.
11	Linear interpolation is not selected in AFGROW.

Forman Material Object - Function Return Codes

Number	Error Description
-1	Unidentified automation error.
0	Normal completion of routine.
1	Rswitched due to R-mapping condition. Not a critical error, function finished.
10	$R_{low\ border}$ value in R-mapping routine was adjusted. Usually, when $R_{low\ border}$ is out of the bounds set by Forman R_{low} and R_{high} . Not a critical error, function finished.
100	$R_{high\ border}$ value in R-mapping routine was adjusted. Usually, when $R_{high\ border}$ is out of the bounds set by Forman R_{lower} and R_{high} . Not a critical error, function finished.
1000	R_{map} value in R-mapping routine was adjusted. Usually when R_{map} is out of the bounds set by Forman R_{low} and R_{high} . Not a critical error, function finished.
1001	R_{cut} value that determine where to apply second data fit was corrected. Usually when R_{cut} is out of the borders set by Forman R_{lower} and R_{higher} . Not critical error, function was finished.

Application Object - Return Codes for Transition Types

Number	Transition Type
0	Transition from Part-Through to Through Crack
1	Transition from Part-Through to Oblique Through Crack
2	Transition from Oblique to Strait Crack
3	Transition from Internal Through to Edge Crack

Application Object - Return Codes for Transition Types

Number	Transition Type
0	Transition based on 95% of the thickness penetration
1	Transition based on 100% of the thickness penetration
2	Transition based on free edge touch (Internal Through Crack)
3	Transition based on K_{max} criteria
4	Transition based on KI_e criteria
5	Transition based on shape criteria (Oblique Crack)

Application Object - Return Codes for Fracture Types

Number	Fracture Type
0	Fracture based on 'Net Section Yield' Criteria (current maximum stress)
1	Fracture based on 'Net Section Yield' Criteria (residual strength requirement)
2	Fracture based on 'Kmax' Criteria (current maximum stress)
3	Fracture based on 'Kmax' Criteria (residual strength requirement)
4	Fracture based on free edge touch

Application Object -Return Codes for GetSpectrumInfo function

Number	Fracture Type
0	Normal completion of the routine
1	Error in the spectrum (sp3) file
2	Unknown subspectrum file type
3	Unable to find subspectrum file
4	Error in the subspectrum file

Appendix 2: AFGROW Constants

AFGROW Models (AfgrowModels)

Value	Name	Description
1000	uPartThrough	Part Through Crack (user-defined)
1010	aCenterSEllipticSurface	Center Semi elliptic Surface Flaw
1015	aCenterSEllipticEdgeSurface	Center Semi elliptic Edge Surface Flaw
1020	aCenterFEllipticEmbedded	Center Full-elliptic Embedded Flaw
1030	aSingleCornerAtHole	Single Corner Crack at Hole
1035	aSingleCornerAtNotch	Single Corner Crack at Notch
1040	aSingleSurfaceAtHole	Single Surface Crack at Hole
1045	aSingleSurfaceAtNotch	Single Surface Crack at Notch
1050	aDoubleCornerAtHole	Double Corner Crack at Hole
1060	aDoubleSurfaceAtHole	Double Surface Crack at Hole
1070	aSingleEdgeCorner	Single Edge Corner Crack
1080	aSingleCornerInLug	Single Corner Crack in Lug
1090	aPartThroughInPipe	Part Through Crack in Pipe
2000	uThrough	Through Crack (user-defined)
2010	aCenterThrough	Center Through Crack
2020	aSingleThroughAtHole	Single Through Crack at Hole
2030	aDoubleThroughAtHole	Double Through Crack at Hole
2035	aSingleThroughCrackAtNotch	Single Through Crack At Notch
2040	aSingleEdgeThrough	Single Edge Through Crack
2050	aDoubleEdgeThrough	Double Edge Through Crack
2060	aWOLCT	WOL/CT Specimen
2070	aSingleEdgeInLug	Single Edge Crack in Lug
2080	aThroughInRod	Rod
2090	aThroughInPipe	Through Crack in Pipe
3010	wCenterSEllipticSurface	Center Semi elliptic Surface Crack
3020	wSingleEdgeCorner	Single Edge Corner Crack
3030	wInternalAxialInThickPipe	Internal Axial Crack in Thick Pipe
3040	wExternalAxialInThickPipe	External Axial Crack in Thick Pipe
4010	wCenterThrough	Center Through Crack
4020	wSingleEdgeThrough	Single Edge Through Crack
4030	wDoubleEdgeThrough	Double Edge Through Crack
4040	wRadialEdgeInDisc	Radial Edge Crack in Disk
4050	wAxialThroughInThickPipe	Axial Through Crack in Thick Pipe

Common Crack Growth Specimen Types (WOL_CTtype)

Value	Name
0	WOL_TYPE

1	CT_TYPE
---	---------

Types of Units (AfgrowUnits)

Value	Name
0	UnitsEnglish
1	UnitsMetric

Crack Growth Directions (CrackDirection)

Value	Name
0	CXdirection
1	AYdirection

Crack Transition Conditions (CrackTransConditions)

Value	Name
0	KIcTransition
1	KIeTransition

Types of Beta Correction Data (TypeOfBetaCorData)

Value	Name
0	NormalizedStress
1	BetaCorrectionFactors

Types of Residual Stress Data (CrackDirection)

Value	Name
0	Stress
1	ResidualK

Residual Stress SIF Table Generation Tyoes (TypeOfResSIFTableGeneration)

Value	Name
0	WeightFunction
1	GaussIntegration

Types of Material Data Available in AFGROW (AfgrowMaterials)

Value	Name
0	MaterialHarterT
1	MaterialWalker
3	MaterialForman
4	MaterialLookup
5	MaterialNASGRO

Types of SIF Data (DataType)

Value	Name
0	NotUserDefined
1	UserDefined

Types of User-Defined Beta Interpolation (Part-Through Cracks)
(TypeOfUserDefBetaPartThroughCrackInt)

Value	Name
0	Linear
1	FourPoints
-1	Error

Types of Retardation Models (AfgrowRetardationModels)

Value	Name
0	RetardNoRetardation
1	RetardWillenborgModel
2	RetardClosureModel
3	RetardWheelerModel

Types of Output Intervals (OutputIntervalType)

Value	Name
0	CrackGrowthIncrement
1	SpectrumCyclicIncrement
2	EachSpectrumStressLevel

Appendix 3: AFGROW Exceptions

Number	Exception Description
28	AFGROW is executing a prediction, cannot change any parameters
29	Cannot open problem definition file
30	Wrong Parameter
31	Unknown Model
32	Cannot set this parameter in AFGROW
33	Wrong file type
34	Beta Correction option in AFGROW can not be used for Weight Function Stress Intensity Solution models
35	This model is not a User defined data model
36	Cannot select material
37	This material data type is not active
38	Retardation model not valid
39	This retardation model is not active

Appendix 4: Sample VBA Source Code (without Events)

```
-----  
-  
Dim mAfgrow As Afgrow.Application  
Dim i As Integer  
  
-----  
-  
Private Sub CommandButton1_Click()  
Dim Cycles As Double  
Dim finalC As Double  
Dim finalA As Double  
Dim finalCt As Double  
Dim finalKc As Double  
Dim finalKa As Double  
Dim finalKct As Double  
Dim result As Boolean  
i = 3  
Set mAfgrow = CreateObject("Afgrow.Application")  
Do While Sheet1.Cells(i, 2) > 0  
    mAfgrow.Model = Sheet1.Cells(i, 6)  
    mAfgrow.SMF = Sheet1.Cells(i, 2)  
    mAfgrow.ConstAmplitudeSpectrum (Sheet1.Cells(i, 3))  
    mAfgrow.CrackLengthA = Sheet1.Cells(i, 4)  
    mAfgrow.CrackLengthC = Sheet1.Cells(i, 5)  
    result = mAfgrow.RunFrozPredict(Cycles, finalC, finalKc, finalA,  
finalKa, finalCt, finalKct)  
    Sheet1.Cells(i, 7) = Cycles  
    Sheet1.Cells(i, 8) = finalC  
    Sheet1.Cells(i + 14, 2) = Cycles  
    Sheet1.Cells(i + 14, 3) = mAfgrow.SMF  
    i = i + 1  
Loop  
mAfgrow.Quit  
Set mAfgrow = Nothing  
End Sub  
  
-----  
-  
Private Sub CommandButton2_Click()  
i = 3  
Do While Sheet1.Cells(i, 2) > 0  
    Sheet1.Cells(i, 7) = 0  
    Sheet1.Cells(i, 8) = 0  
    i = i + 1  
Loop  
End Sub
```

Appendix 5: Sample VBA Source Code (with Events)

```
Public WithEvents mAfgrow As Afgrow.Application
Public i As Integer

-----

Private Sub mAfgrow_PredictFinished(ByVal result As Integer, ByVal
dblCycles As Double, ByVal dblKc As Double, ByVal dblKa As Double,
ByVal dblKct As Double, ByVal dblC As Double, ByVal dblA As Double,
ByVal dblCt As Double)
Sheet1.Cells(i, 7) = dblCycles
i = i + 1
If Sheet1.Cells(i, 2) > 0 Then
    Call Run
Else:
    mAfgrow.Quit
    Set mAfgrow = Nothing
End If
End Sub

-----

Private Sub CommandButton1_Click()
i = 6
Set mAfgrow = CreateObject("afgrow.application")
mAfgrow.Visible = True
Call Run
End Sub

-----

Public Sub Run()
mAfgrow.ConstAmplitudeSpectrum (Sheet1.Cells(i, 3))
mAfgrow.Model = Sheet1.Cells(i, 4)
mAfgrow.SMF = Sheet1.Cells(i, 2)
mAfgrow.CrackLengthA = Sheet1.Cells(i, 5)
mAfgrow.CrackLengthC = Sheet1.Cells(i, 6)
mAfgrow.RunPredict
End Sub
```

Appendix 6: Sample VBA Source Code (Using Predict Preferences)

```
Public WithEvents mAfgrow As Afgrow.Application
Public prefs As PredictPreferences
Public i As Integer
Public j As Integer
Dim SpectrumFile As String
Dim answer As Boolean

-----
Private Sub mAfgrow_PredictFinished(ByVal result As Integer, ByVal
dblCycles As Double, ByVal dblKc As Double, ByVal dblKa As Double,
ByVal dblKct As Double, ByVal dblC As Double, ByVal dblA As Double,
ByVal dblCt As Double)
Sheet1.Cells(i, 7) = dblCycles

If result = 0 Then
    Sheet1.Cells(i, 8) = "Completed Normally"
ElseIf result = 1 Then
    Sheet1.Cells(i, 8) = "After one pass of the spectrum, growth was less
than 1.0e-13 Program halted"
ElseIf result = 2 Then
    Sheet1.Cells(i, 8) = "Maximum Number of passes exceeded"
ElseIf result = 3 Then
    Sheet1.Cells(i, 8) = "Crack Length Exceeded Stop Value"
ElseIf result = 4 Then
    Sheet1.Cells(i, 8) = "Cycle Count Exceeded Stop Value"
ElseIf result = 10 Then
    Sheet1.Cells(i, 8) = "No Spectrum file specified... Can not Predict"
ElseIf result = 11 Then
    Sheet1.Cells(i, 8) = "Beta table has zero length... Can not Predict"
ElseIf result = 12 Then
    Sheet1.Cells(i, 8) = "Repair patch applied and initiation on... Can
not Predict! "
ElseIf result = 13 Then
    Sheet1.Cells(i, 8) = "Initiation is not allowed with this model. Can
not Predict! "
ElseIf result = 14 Then
    Sheet1.Cells(i, 8) = "NASGRO equation Rhi value is to large"
ElseIf result = 20 Then
    Sheet1.Cells(i, 8) = "Wrong geometry for an oblique crack"
ElseIf result = 21 Then
    Sheet1.Cells(i, 8) = "The initial crack length in the thickness
direction is greater than the input thickness. Check data.."
ElseIf result = 22 Then
    Sheet1.Cells(i, 8) = "Error in spectrum file(s)"
ElseIf result = 23 Then
    Sheet1.Cells(i, 8) = "Unable to open plotfile"
ElseIf result = 24 Then
    Sheet1.Cells(i, 8) = "Error in Initiation. Predict stopped"
ElseIf result = 25 Then
    Sheet1.Cells(i, 8) = "The number of cycles to initiation is greater
than 2.e+9. Predict stopped"
ElseIf result = 26 Then
    Sheet1.Cells(i, 8) = "Not enough memory to allocate spectrum"
ElseIf result = 27 Then
    Sheet1.Cells(i, 8) = "Only a BLOCKED spectrum may have more than one
(1) cycle per stress level"
ElseIf result = 28 Then
```

```

    Sheet1.Cells(i, 8) = "Program termination by user"
ElseIf result = -1 Then
    Sheet1.Cells(i, 8) = "Unknown Error"
End If

i = i + 1
If Sheet1.Cells(i, 2) > 0 Then
    CurrentBox.Text = CStr(i)
    Call Run
Else:
    answer = mAfgrow.Quit
    Set mAfgrow = Nothing
    Unload UserForm1
End If
End Sub

```

```

-----
Private Sub CommandButton1_Click()
i = CInt(StartBox.Text)
CurrentBox.Text = CStr(i)
mAfgrow.Visible = True
Call Run
End Sub

```

```

-----
Private Sub CommandButton2_Click()
j = 6
Do While Sheet1.Cells(j, 2) > 0
    Sheet1.Cells(j, 7) = 0
    Sheet1.Cells(j, 8) = " "
    j = j + 1
Loop
End Sub

```

```

-----
Private Sub CommandButton3_Click()
    answer = mAfgrow.Quit
    Set mAfgrow = Nothing
    Unload UserForm1
End Sub

```

```

-----
Private Sub UserForm_Initialize()
SpectrumFile = Sheet1.Cells(3, 2)
Set mAfgrow = CreateObject("Afgrow.Application")
answer = mAfgrow.OpenSpectrumFile(SpectrumFile)
Set prefs = mAfgrow.PredictPreferences
Call prefs.SetPropagationLimits(True, , , , 1.5) ← "Call" is used since
this function returns a value
prefs.bOutputToFile = True
End Sub

```

```

-----
Public Sub Run()
mAfgrow.Model = Sheet1.Cells(i, 3)

```

```
mAfgrow.SMF = Sheet1.Cells(i, 2)
mAfgrow.CrackLengthA = Sheet1.Cells(i, 4)
mAfgrow.CrackLengthC = Sheet1.Cells(i, 5)
prefs.sOutputFileName = Sheet1.Cells(i, 6)
answer = mAfgrow.RunPredict
End Sub
```

Appendix 7: Sample VBA Source Code (Using Beta Correction Tables)

```
Public WithEvents mAfgrow As Afgrow.Application
Public prefs As PredictPreferences
Public betacorr As BetaCorrection
Public i As Integer
Public j As Integer
Dim CArray(4, 2) As Double
Dim SpectrumFile As String
Dim Middle As Variant
Dim answer As Boolean

-----

Private Sub mAfgrow_PredictFinished(ByVal result As Integer, ByVal
dblCycles As Double, ByVal dblKc As Double, ByVal dblKa As Double,
ByVal dblKct As Double, ByVal dblC As Double, ByVal dblA As Double,
ByVal dblCt As Double)
Sheet1.Cells(i, 7) = dblCycles

If result = 0 Then
    Sheet1.Cells(i, 8) = "Completed Normally"
ElseIf result = 1 Then
    Sheet1.Cells(i, 8) = "After one pass of the spectrum, growth was less
than 1.0e-13 Program halted"
ElseIf result = 2 Then
    Sheet1.Cells(i, 8) = "Maximum Number of passes exceeded"
ElseIf result = 3 Then
    Sheet1.Cells(i, 8) = "Crack Length Exceeded Stop Value"
ElseIf result = 4 Then
    Sheet1.Cells(i, 8) = "Cycle Count Exceeded Stop Value"
ElseIf result = 10 Then
    Sheet1.Cells(i, 8) = "No Spectrum file specified... Can not Predict"
ElseIf result = 11 Then
    Sheet1.Cells(i, 8) = "Beta table has zero length... Can not Predict"
ElseIf result = 12 Then
    Sheet1.Cells(i, 8) = "Repair patch applied and initiation on... Can
not Predict! "
ElseIf result = 13 Then
    Sheet1.Cells(i, 8) = "Initiation is not allowed with this model. Can
not Predict! "
ElseIf result = 14 Then
    Sheet1.Cells(i, 8) = "NASGRO equation Rhi value is to large"
ElseIf result = 20 Then
    Sheet1.Cells(i, 8) = "Wrong geometry for an oblique crack"
ElseIf result = 21 Then
    Sheet1.Cells(i, 8) = "The initial crack length in the thickness
direction is greater than the input thickness. Check data.."
ElseIf result = 22 Then
    Sheet1.Cells(i, 8) = "Error in spectrum file(s)"
ElseIf result = 23 Then
    Sheet1.Cells(i, 8) = "Unable to open plotfile"
ElseIf result = 24 Then
    Sheet1.Cells(i, 8) = "Error in Initiation. Predict stopped"
ElseIf result = 25 Then
    Sheet1.Cells(i, 8) = "The number of cycles to initiation is greater
than 2.e+9. Predict stopped"
```

```

ElseIf result = 26 Then
    Sheet1.Cells(i, 8) = "Not enough memory to allocate spectrum"
ElseIf result = 27 Then
    Sheet1.Cells(i, 8) = "Only a BLOCKED spectrum may have more than one
(1) cycle per stress level"
ElseIf result = 28 Then
    Sheet1.Cells(i, 8) = "Program termination by user"
ElseIf result = -1 Then
    Sheet1.Cells(i, 8) = "Unknown Error"
End If

i = i + 1
If Sheet1.Cells(i, 2) > 0 Then
    CurrentBox.Text = CStr(i)
    Call Run
Else:
    answer = mAfgrow.Quit
    Set mAfgrow = Nothing
    Unload UserForm1
End If
End Sub

```

```

Private Sub CommandButton1_Click()
i = CInt(StartBox.Text)
CurrentBox.Text = CStr(i)
mAfgrow.Visible = True
Call Run
End Sub

```

```

Private Sub CommandButton2_Click()
j = 6
Do While Sheet1.Cells(j, 2) > 0
    Sheet1.Cells(j, 7) = 0
    Sheet1.Cells(j, 8) = " "
    j = j + 1
Loop
End Sub

```

```

Private Sub CommandButton3_Click()
    answer = mAfgrow.Quit
    Set mAfgrow = Nothing
    Unload UserForm1
End Sub

```

```

Private Sub UserForm_Initialize()
Dim k As Integer
SpectrumFile = Sheet1.Cells(3, 2)
Set mAfgrow = CreateObject("Afgrow.Application")
answer = mAfgrow.OpenSpectrumFile(SpectrumFile)

```



```

Set prefs = mAfgrow.PredictPreferences
Call prefs.SetPropagationLimits(True, , , , 1.5)
prefs.bOutputToFile = True
Set betacorr = mAfgrow.BetaCorrection
mAfgrow.Visible = False
End Sub

```

```

Public Sub Run()
mAfgrow.Model = Sheet1.Cells(i, 3)
betacorr.bUseBetaCorrection = True
betacorr.nTypeOfData = BetaCorrectionFactors
k = 0
Do While k < 5
  CArray(k, 0) = Sheet1.Cells(16 + k, 2)
  CArray(k, 1) = Sheet1.Cells(16 + k, 3)
  CArray(k, 2) = Sheet1.Cells(16 + k, 4)
  k = k + 1
Loop
mAfgrow.Model = aSingleCornerAtHole
Middle = CArray
betacorr.aBetaCorrectionData = Middle
mAfgrow.SMF = Sheet1.Cells(i, 2)
mAfgrow.CrackLengthA = Sheet1.Cells(i, 4)
mAfgrow.CrackLengthC = Sheet1.Cells(i, 5)
prefs.sOutputFileName = Sheet1.Cells(i, 6)
answer = mAfgrow.RunPredict
End Sub

```

Note: Arrays in visual BASIC begin with element "zero," not "one." Arrays dimensioned as Array(4) actually contain 5 elements (0,1,2,3,4)

Appendix 8: Sample VBA Source Code (User-Defined Betas and Tabular Crack Growth Rate)

```
Public WithEvents mAfgrow As Afgrow.Application
Public prefs As PredictPreferences
Public user_beta As UserDefinedBeta
Public table_lookup As TabularLookupMaterial
Public i As Integer
Public j As Integer
Public k As Integer
Dim AArray(2) As Double
Dim CArray(2) As Double
Dim BaArray(2, 2) As Double
Dim BcArray(2, 2) As Double
Dim BTable(5, 1) As Double
Dim KTable(25, 2) As Double
Dim SpectrumFile As String
Dim beta_result As Integer
Dim answer As Boolean

Private Sub mAfgrow_PredictFinished(ByVal result As Integer, ByVal
dblCycles As Double, ByVal dblKc As Double, ByVal dblKa As Double,
ByVal dblKct As Double, ByVal dblC As Double, ByVal dblA As Double,
ByVal dblCt As Double)
Sheet1.Cells(i, 6) = dblCycles

If result = 0 Then
    Sheet1.Cells(i, 7) = "Completed Normally"
ElseIf result = 1 Then
    Sheet1.Cells(i, 7) = "After one pass of the spectrum, growth was less
than 1.0e-13 Program halted"
ElseIf result = 2 Then
    Sheet1.Cells(i, 7) = "Maximum Number of passes exceeded"
ElseIf result = 3 Then
    Sheet1.Cells(i, 7) = "Crack Length Exceeded Stop Value"
ElseIf result = 4 Then
    Sheet1.Cells(i, 7) = "Cycle Count Exceeded Stop Value"
ElseIf result = 10 Then
    Sheet1.Cells(i, 7) = "No Spectrum file specified... Can not Predict"
ElseIf result = 11 Then
    Sheet1.Cells(i, 7) = "Beta table has zero length... Can not Predict"
ElseIf result = 12 Then
    Sheet1.Cells(i, 7) = "Repair patch applied and initiation on... Can
not Predict! "
ElseIf result = 13 Then
    Sheet1.Cells(i, 7) = "Initiation is not allowed with this model. Can
not Predict! "
ElseIf result = 14 Then
    Sheet1.Cells(i, 7) = "NASGRO equation Rhi value is to large"
ElseIf result = 20 Then
    Sheet1.Cells(i, 7) = "Wrong geometry for an oblique crack"
ElseIf result = 21 Then
    Sheet1.Cells(i, 7) = "The initial crack length in the thickness
direction is greater than the input thickness. Check data.."
ElseIf result = 22 Then
    Sheet1.Cells(i, 7) = "Error in spectrum file(s)"
ElseIf result = 23 Then
```

```

Sheet1.Cells(i, 7) = "Unable to open plotfile"
ElseIf result = 24 Then
    Sheet1.Cells(i, 7) = "Error in Initiation. Predict stopped"
ElseIf result = 25 Then
    Sheet1.Cells(i, 7) = "The number of cycles to initiation is greater
than 2.e+9. Predict stopped"
ElseIf result = 26 Then
    Sheet1.Cells(i, 7) = "Not enough memory to allocate spectrum"
ElseIf result = 27 Then
    Sheet1.Cells(i, 7) = "Only a BLOCKED spectrum may have more than one
(1) cycle per stress level"
ElseIf result = 28 Then
    Sheet1.Cells(i, 7) = "Program termination by user"
ElseIf result = -1 Then
    Sheet1.Cells(i, 7) = "Unknown Error"
End If

i = i + 1
If Sheet1.Cells(i, 2) > 0 Then
    CurrentBox.Text = CStr(i)
    Call Run
Else:
    answer = mAfgrow.Quit
    Set mAfgrow = Nothing
    Unload UserForm1
End If
End Sub

Private Sub CommandButton1_Click()
i = CInt(StartBox.Text)
CurrentBox.Text = CStr(i)
Call Run
End Sub

Private Sub CommandButton2_Click()
j = 6
Do While Sheet1.Cells(j, 2) > 0
    Sheet1.Cells(j, 6) = 0
    Sheet1.Cells(j, 7) = " "
    j = j + 1
Loop
End Sub

Private Sub CommandButton3_Click()
    answer = mAfgrow.Quit
    Set mAfgrow = Nothing
    Unload UserForm1
End Sub

Private Sub UserForm_Initialize()
Dim k As Integer
SpectrumFile = Sheet1.Cells(3, 2)
Set mAfgrow = CreateObject("Afgrow.Application")
answer = mAfgrow.OpenSpectrumFile(SpectrumFile)
Set prefs = mAfgrow.PredictPreferences
Call prefs.SetPropagationLimits(True, , , , 1.5)
prefs.bOutputToFile = True
Set user_beta = mAfgrow.UserDefinedBeta
Set table_lookup = mAfgrow.SetTabularLookupMaterial
mAfgrow.Visible = False

```

```

'set the user-defined beta information
mAfgrow.Model = uPartThrough
j = 0
Do While j < 3
CArray(j) = Sheet2.Cells(j + 5, 2)
k = 0
Do While k < 3
AArray(k) = Sheet2.Cells(4, k + 3)
BaArray(j, k) = Sheet2.Cells(j + 5, k + 3)
BcArray(j, k) = Sheet2.Cells(j + 10, k + 3)
k = k + 1
Loop
j = j + 1
Loop
j = 0
Do While j < 6
BTArray(j, 0) = Sheet2.Cells(j + 16, 2)
BTArray(j, 1) = Sheet2.Cells(j + 16, 3)
j = j + 1
Loop
beta_result = user_beta.SetLineaInt(AArray, CArray, BaArray)
Dim middle As Variant
middle = BTArray
user_beta.aThroughCrackData = middle

```

The variable, beta_result is used here in case a user would want to check the contents of the value returned by this function. The "Call" option could also have been used if there was no desire to save the returned error code.

'Set the crack growth rate data using tabular look-up

```

k = 0
Do While k < 3
j = 0
Do While j < 26
If (k = 0 And j = 0) Then
KTable(j, k) = 0#
Else
KTable(j, k) = Sheet3.Cells(j + 2, k + 2)
End If
j = j + 1
Loop
k = k + 1
Loop
table_lookup.sMaterialName = Sheet3.Cells(2, 6)
Dim kVar As Variant
kVar = KTable
Call table_lookup.SetData(kVar, Sheet3.Cells(8, 7), Sheet3.Cells(8, 6),
Sheet3.Cells(5, 7), Sheet3.Cells(5, 6), Sheet3.Cells(8, 9))
table_lookup.dKc = Sheet3.Cells(5, 8)
table_lookup.dKIc = Sheet3.Cells(5, 9)
table_lookup.dPoissonsRatio = Sheet3.Cells(5, 10)
table_lookup.dThermalEx = Sheet3.Cells(8, 8)
table_lookup.dYield = Sheet3.Cells(8, 10)
table_lookup.dYoungModulus = Sheet3.Cells(8, 11)
End Sub

```

The tabular crack-growth rate data array is setup to be in the same form as AFGROW uses in the paste to Excel option. The (0,0) array element is not used by AFGROW, but is used as a label (dadn/R) to let users know that the first column contains the da/dn values and the first row contains the stress ratio (R) data. BE SURE to look at the details for the tabular lookup interface in section 11.

```

Public Sub Run()
mAfgrow.SMF = Sheet1.Cells(i, 2)
mAfgrow.CrackLengthA = Sheet1.Cells(i, 3)
mAfgrow.CrackLengthC = Sheet1.Cells(i, 4)
prefs.sOutputFileName = Sheet1.Cells(i, 5)
answer = mAfgrow.RunPredict
End Sub

```

— THIS PAGE WAS INTENTIONALLY LEFT BLANK

Appendix 9: Sample VBA Source Code (Using Predict Info)

```
Public WithEvents mAfgrow As Afgrow.Application
Public WithEvents mAfgrow As Afgrow.Application
Public prefs As PredictPreferences
Public user_beta As UserDefinedBeta
Public table_lookup As TabularLookupMaterial
Public i As Integer
Public j As Integer
Public k As Integer
Dim Count As Integer
Dim AArray(2) As Double
Dim CArray(2) As Double
Dim BaArray(2, 2) As Double
Dim BcArray(2, 2) As Double
Dim BtArray(5, 1) As Double
Dim KTable(25, 2) As Double
Dim SpectrumFile As String
Dim beta_result As Integer
Dim answer As Boolean
Dim NewRun As Boolean
```

PredictFinished is an event that belongs to the AFGROW Application. That is why it is associated with the variable – mAfgrow. The PredictFinished event is found in the right hand pull-down menu (event list) when the application item (m_Afgrow) is selected in the left hand pull-down menu.

```
Private Sub mAfgrow_PredictFinished(ByVal result As Integer, ByVal
dblCycles As Double, ByVal dblKc As Double, ByVal dblKa As Double,
ByVal dblKct As Double, ByVal dblC As Double, ByVal dblA As Double,
ByVal dblCt As Double)
Sheet1.Cells(i, 6) = dblCycles

If result = 0 Then
    Sheet1.Cells(i, 7) = "Completed Normally"
ElseIf result = 1 Then
    Sheet1.Cells(i, 7) = "After one pass of the spectrum, growth was less
than 1.0e-13 Program halted"
ElseIf result = 2 Then
    Sheet1.Cells(i, 7) = "Maximum Number of passes exceeded"
ElseIf result = 3 Then
    Sheet1.Cells(i, 7) = "Crack Length Exceeded Stop Value"
ElseIf result = 4 Then
    Sheet1.Cells(i, 7) = "Cycle Count Exceeded Stop Value"
ElseIf result = 10 Then
    Sheet1.Cells(i, 7) = "No Spectrum file specified... Can not Predict"
ElseIf result = 11 Then
    Sheet1.Cells(i, 7) = "Beta table has zero length... Can not Predict"
ElseIf result = 12 Then
    Sheet1.Cells(i, 7) = "Repair patch applied and initiation on... Can
not Predict! "
ElseIf result = 13 Then
    Sheet1.Cells(i, 7) = "Initiation is not allowed with this model. Can
not Predict! "
ElseIf result = 14 Then
    Sheet1.Cells(i, 7) = "NASGRO equation Rhi value is to large"
ElseIf result = 20 Then
    Sheet1.Cells(i, 7) = "Wrong geometry for an oblique crack"
ElseIf result = 21 Then
    Sheet1.Cells(i, 7) = "The initial crack length in the thickness
direction is greater than the input thickness. Check data.."
ElseIf result = 22 Then
    Sheet1.Cells(i, 7) = "Error in spectrum file(s)"
```

```

ElseIf result = 23 Then
    Sheet1.Cells(i, 7) = "Unable to open plotfile"
ElseIf result = 24 Then
    Sheet1.Cells(i, 7) = "Error in Initiation. Predict stopped"
ElseIf result = 25 Then
    Sheet1.Cells(i, 7) = "The number of cycles to initiation is greater
than 2.e+9. Predict stopped"
ElseIf result = 26 Then
    Sheet1.Cells(i, 7) = "Not enough memory to allocate spectrum"
ElseIf result = 27 Then
    Sheet1.Cells(i, 7) = "Only a BLOCKED spectrum may have more than one
(1) cycle per stress level"
ElseIf result = 28 Then
    Sheet1.Cells(i, 7) = "Program termination by user"
ElseIf result = -1 Then
    Sheet1.Cells(i, 7) = "Unknown Error"
End If

```

```

Sheet4.Cells(1, 7) = Count
i = i + 1
If Sheet1.Cells(i, 2) > 0 Then
    CurrentBox.Text = CStr(i)
    Call Run
Else:
    answer = mAfgrow.Quit
    Set mAfgrow = Nothing
    Unload UserForm1
End If
End Sub

```

Write the number of output lines above the data

```

Private Sub CommandButton1_Click()
i = CInt(StartBox.Text)
CurrentBox.Text = CStr(i)
Call Run
End Sub

```

```

Private Sub CommandButton2_Click()
j = 6
Do While Sheet1.Cells(j, 2) > 0
    Sheet1.Cells(j, 6) = 0
    Sheet1.Cells(j, 7) = " "
    j = j + 1
Loop
End Sub

```

```

Private Sub CommandButton3_Click()
    answer = mAfgrow.Quit
    Set mAfgrow = Nothing
    Unload UserForm1
End Sub

```

PredictInfo is also an event that belongs to the AFGROW Application. That is why it is associated with the variable - mAfgrow. The PredictInfo event is found in the right hand pull-down menu (event list) when the application item (m_Afgrow) is selected in the left hand pull-down menu.

```

Private Sub mAfgrow_PredictInfo(ByVal Model As Long, ByVal dStress As
Double, ByVal dRStress As Double, ByVal dCycles As Double, ByVal dPass
As Long, ByVal iOutputInfo1 As Object, ByVal iOutputInfo2 As Object,
ByVal iOutputInfo3 As Object, ByVal iOutputInfo4 As Object)
Dim iOutput1 As Afgrow.OutputInfoInt
Dim iOutput2 As Afgrow.OutputInfoInt
Set iOutput1 = iOutputInfo1
If Model < 2000 Then Set iOutput2 = iOutputInfo2
If NewRun = True Then

```

Define the output objects - In this case, for the "C" and "A" direction. Remember, if the model code is < 2000, ONLY if there is an "A" direction.

```

j = 1
Do While ((j < 500) Or (Sheet4.Cells(j + 1, 1) = " "))
    Sheet4.Cells(j + 1, 1) = " "
    Sheet4.Cells(j + 1, 2) = " "
    Sheet4.Cells(j + 1, 3) = " "
    Sheet4.Cells(j + 1, 4) = " "
    Sheet4.Cells(j + 1, 5) = " "
    j = j + 1
Loop
Count = 0
End If
NewRun = False
Count = Count + 1
Sheet4.Cells(Count + 1, 1) = dCycles
Sheet4.Cells(Count + 1, 2) = iOutput1.dCrackLength
Sheet4.Cells(Count + 1, 3) = iOutput1.dBeta
If Model < 2000 Then
    Sheet4.Cells(Count + 1, 4) = iOutput2.dCrackLength
    Sheet4.Cells(Count + 1, 5) = iOutput2.dBeta
End If
Set iOutput1 = Nothing
End Sub

Private Sub UserForm_Activate()
Dim k As Integer
SpectrumFile = Sheet1.Cells(3, 2)
Set mAfgrow = CreateObject("Afgrow.Application")
answer = mAfgrow.OpenSpectrumFile(SpectrumFile)
Set prefs = mAfgrow.PredictPreferences
prefs.bOutputToFile = True
Set user_beta = mAfgrow.UserDefinedBeta
Set table_lookup = mAfgrow.SetTabularLookupMaterial
mAfgrow.Visible = False

'set the user-defined beta information
mAfgrow.Model = uPartThrough
j = 0
Do While j < 3
    CArray(j) = Sheet2.Cells(j + 5, 2)
    k = 0
    Do While k < 3
        AArray(k) = Sheet2.Cells(4, k + 3)
        BaArray(j, k) = Sheet2.Cells(j + 5, k + 3)
        BcArray(j, k) = Sheet2.Cells(j + 10, k + 3)
        k = k + 1
    Loop
    j = j + 1
Loop
j = 0
Do While j < 6
    BTAArray(j, 0) = Sheet2.Cells(j + 16, 2)
    BTAArray(j, 1) = Sheet2.Cells(j + 16, 3)
    j = j + 1
Loop
beta_result = user_beta.SetLineaInt(AArray, CArray, BaArray, BcArray)
Dim middle As Variant
middle = BTAArray
user_beta.aThroughCrackData = middle

'Set the crack growth rate data using tabular look-up

```

Clears the data from the previous analysis

In this case, only the cycles, crack length, and beta information is being written to the spreadsheet.


```

k = 0
Do While k < 3
j = 0
Do While j < 26
If (k = 0 And j = 0) Then
    KTable(j, k) = 0#
Else
    KTable(j, k) = Sheet3.Cells(j + 2, k + 2)
End If
j = j + 1
Loop
k = k + 1
Loop
table_lookup.sMaterialName = Sheet3.Cells(2, 6)
Dim kVar As Variant
kVar = KTable
Call table_lookup.SetData(kVar, Sheet3.Cells(8, 7), Sheet3.Cells(8, 6),
Sheet3.Cells(5, 7), Sheet3.Cells(5, 6), Sheet3.Cells(8, 9))
table_lookup.dKc = Sheet3.Cells(5, 8)
table_lookup.dKlc = Sheet3.Cells(5, 9)
table_lookup.dPoissonsRatio = Sheet3.Cells(5, 10)
table_lookup.dThermalEx = Sheet3.Cells(8, 8)
table_lookup.dYield = Sheet3.Cells(8, 10)
table_lookup.dYoungModulus = Sheet3.Cells(8, 11)

End Sub

Public Sub Run()
mAfgrow.SMF = Sheet1.Cells(i, 2)
mAfgrow.CrackLengthA = Sheet1.Cells(i, 3)
mAfgrow.CrackLengthC = Sheet1.Cells(i, 4)
prefs.sOutputFileName = Sheet1.Cells(i, 5)
NewRun = True
answer = mAfgrow.RunPredict
End Sub

```

Appendix 10: Material Codes

Harter T Material Data Codes

Material; Condition; Environment*	Code
2024 T-3 LONG CRACK DATA	1
2024-T851 - LONG CRACK	2
6061-T6511 EXTRUSION	3
7050 T74 PLATE	4
7075-T6511 EXTRUSION	5
7075-T73 L-T FORGING	6
7075-T73 L-T [DRY AIR]	7
7075-T73 T-L FORGING	8
Ti-6-4 ALPHA-BETA FORGING	9
Ti-6-4 AMS-4911G ANNEALED	10
4340-150 KSI FORGING	11
4340-180 KSI FORGING	12

NASGRO Material Data Codes

Material; Condition; Environment*	Code
[A] Iron, alloy or cast	
ASTM Specification	
A536 Grd 80-55-06	
As cast	A1AC50AB1
[B] ASTM spec. grd. Steel	
A10 Series	
A36	
Plt(Dyn Klc, < 500Hz); LA, HHA, 3% NaCl	B0CB10AB1
ES Weld & HAZ(Dyn Klc, < 500Hz); LA,HHA, 3% NaCl	B0CZK1AB1
A200 Series	
A203 Grd E (3.5% Ni)	
Plt	B2CE12AB1
Plt; -100F	B2CE12LA7
A216 Grd WCC	
Casting	B2GC51AB1
A300 Series	
A302 Grd B	
Plt	B3AB12AB1
A372 Type IV	
Forg	B3GD21AB1
A387 Grd 22, Cl 2	
Plt	B3IQ10AB1

* Unless noted, assume Lab Air (LA) environment and any orientation except S-T, S-L, C-R, C-L, and R-L.

<u>A400 Series</u>	
A469 CI 4	
Forg	B4JD26AB1
A469 CI 5	
Forg	B4JE20AB1
A500 Series	
A508 CI2 & CI3	
Forg	B5AC21AB1
A514 Typ F	
Plt	B5BF10AB1
GMA SR Weld	B5BFC2AB1
A517 Grd F (T1 Steel)	
Plt	B5DF12AB1
A533-B, CI1 & CI2	
Plt	B5HD10AB1
SMA Weld	B5HDF1AB1
A553 Typ I	
Plt	B5QA12AB1
Plt; -320F	B5QA12LA4
<u>[B] ASTM spec. grade steel</u>	
A500 Series	
A579 Grd 75 (12% Ni)	
Forg	B5VW20AB1
A588 Grd A & Grd B	
Plt	B5XA11AB1
Plt; 3% NaCl, >0.2 Hz	B5XA11WB1
A645 (5% Ni)	
Plt	B6GA12AB1
Plt; -320F	B6GA12LA4
<u>[C] AISI - SAE Steel</u>	
<u>AISI 10xx-12xx Steel</u>	
Low Carbon 1005-1012	
Hot rolled plt	C1AB11AB1
Low Carbon 1015-1025	
Hot rolled plt	C1BB11AB1
<u>AISI 43xx-48xx Steel</u>	
4330V MOD	
180-200 UTS; Plt & Forg	C4BS10AB1
200-220 UTS; Plt & Forg	C4BT10AB1
220-240 UTS; Plt & Forg	C4BU10AB1
4340	
160-180 UTS; Plt & Forg	C4DC21AB1
180-200 UTS; Plt & Forg; HHA	C4DD11AD1
180-200 UTS; Plt & Forg	C4DD21AB1
200-220 UTS; Plt & Forg	C4DE11AB1

200-220 UTS; Plt & Forg; -50F	C4DE11LB7
220-240 UTS; Plt & Forg	C4DF11AB1
220-240 UTS; Plt & Forg; -50F	C4DF11LB7
240-280 UTS; Plt & Forg	C4DG11AB1
240-280 UTS; Plt & Forg; -50F	C4DG11LB7
[D] Misc. U.S. Spec. Grade Steel	
SAE Spec Steel	
0030 Cast	D5AC50AB1
[E] Trade/common name steel	
Ultra High Strength Steel	
18 Ni Maraging	
250 Grd; Plt & Forg	E1AD10AB1
300 Grd; Plt & Forg	E1AE10AB1
300M	
270-300 UTS; Plt & Forg	E1BF21AB1
AF1410	
220-240 UTS; Plt & Forg; -65F	E1CC12AA7
220-240 UTS; Plt & Forg; LA, HHA / DW > 1Hz	E1CC12AB1
D6AC	
220-240 UTS; Plt & Forg; Nom. Klc(70); -40F	E1DD10AA8
220-240 UTS; Plt & Forg; Nom. Klc (70)	E1DD10AB1
220-240 UTS; Plt & Forg; Nom. Klc(70); HHA/DW>0.1 Hz	E1DD10AD1
220-240 UTS; Plt & Forg; High Klc (90)	E1DJ10AB1
HP-9-4-20	
190-210 UTS; Plt & Forg; L-T, T-L; HHA, SW > 1 Hz	E1EB23AB1
190-210 UTS; Plt & Forg; L-T, T-L; -65F	E1EB23AC7
190-210 UTS; GTA Weld + SR; LA, HHA, SW > 1 Hz	E1ECB2WA1
190-210 UTS; GTA Weld + SR; -65F	E1ECB2AC7
HP-9-4-30	
220-240 UTS; Plt & Forg; L-T, T-L; LA, HHA, SW > 1Hz	E1GC23AB1
220-240 UTS; Plt & Forg; L-T, T-L; -65F	E1GC23AA7
220-240 UTS; Plt & Forg; L-T, T-L; 600F	E1GC23AA14
HY-180(10Ni)	
Plt & Forg	E1IB13AB1
Plt & Forg; DW, ASW > 0.1 Hz	E1IB13WA1
Plt & Forg; SW > 0.1 Hz	E1IB13WB1
HY-TUF	
220-240 UTS; VAR Forg	E1JB23AB1
H-11 MOD	
240-260 UTS; Plt & Forg	E1LE23AB1
[E] Trade/Common Name Steel	
Pressure Vessel / Piping	
HY 80	
Plt	E2AA13AB1
Plt; 3.5% NaCl / SW > 0.1 Hz	E2AA13WB1

HY 130	
Plt	E2CA13AB1
Plt; 3.5% NaCl / SW > 0.1 Hz	E2CA13WB1
GMA Weld	E2CAC1AB1
SMA Weld	E2CAF1AB1
Construction Grade	
HT-80	
Plt	E3BA13AB1
SA Weld	E3BAH1AB1
[F] AISI type stainless steel	
AISI 300 Series	
AISI 301/302	
Ann Plt & Sht	F3AA13AB1
1/2 Hard sht	F3AC13AB1
Full Hard sht	F3AE13AB1
AISI 304/304L	
Ann Plt & Sht, Cast; 550F Air	F3DA13AA13
Ann Plt & Sht, Cast; 800F Air, >1Hz	F3DA13AA16
Ann Plt & Sht, Cast	F3DA13AB1
Ann Plt & Sht, Cast; -320F LN2	F3DA13LA4
SA weld (308 filler) + SR; 800F Air, >1 Hz	F3DAH2AA16
SA weld (308 filler) + SR	F3DAH2AB1
AISI 316/316L	
Ann Plt & Sht, Cast; 600F Air	F3KA13AA14
Ann Plt & Sht, Cast; 800F Air	F3KA13AA16
Ann Plt & Sht, Cast	F3KA13AB1
Ann Plt & Sht, Cast; -452F Lhe	F3KA13LA2
Ann Plt & Sht, Cast; -320F LN2	F3KA13LA4
Cast; 600F Air	F3KA50AA14
Cast	F3KA50AB1
Cast; -453F Lhe	F3KA50LA2
Cast; -320F LN2	F3KA50LA4
SMA weld (316 filler) + SR; 800F Air, >1Hz	F3KAH2AA16
20% CW Plt & Sht	F3KB13AB1
AISI 400 Series	
AISI 430 VAR	
Ann Rnd, C-R	F4LA16AB1
AISI 440C Steel	
Single temper 450F/2hr, T-L	F4SE12AB1
[G] Misc. CRES/heat resistant steel	
PHxx-x Alloys	
PH13-8Mo	
H1000; Plt, Forg, Extr	G1AD13AB1
H1000; Plt & Forg; DW & SW, >1Hz	G1AD13WD1
H1050; Plt & Forg	G1AF13AB1
H1050; Plt & Forg; DW & SW, >0.1Hz	G1AF13WD1

<u>xx-xPH Alloys</u>	
15-5PH	
H900; Rnd, C-R	G2AB16AB1
H1025; Rnd, C-R	G2AD16AB1
H1025; Forg	G2AD23AB1
H1100; Rnd, C-R	G2AF16AB1
17-4PH	
H900; Plt, L-T	G2CB11AB1
H900; Plt, T-L	G2CB12AB1
H1050; Plt	G2CE13AB1
H1025; Rnd, C-L	G2CE19AB1
H1025; Cast; HHA	G2CE50AD1
H1100; Plt; HHA	G2CH13AD1
17-7PH	
TH1050; Plt	G2EH13AB1
<u>AMxxx Alloys</u>	
AM 350	
CRT; Sht, L-T	G4AH11AB1
AM 367	
SCT(850); Sht	G4FC11AB1
<u>Custom xxx Alloys</u>	
Custom 455	
H1000; Plt & Forg	G5BD23AB1
H1025; Forg, C-R	G5BE26AB1
<u>Nitronic xx Alloys</u>	
Nitronic 33	
Ann; Plt	G7AA13AB1
Ann; Plt; -452F Lhe	G7AA13LA2
Ann; Plt; -320F LN2	G7AA13LA4
Nitronic 50	
Ann; Plt	G7CA13AB1
Ann; Plt; -452F Lhe	G7CA13LA2
Ann; Plt; -320F LN2	G7CA13LA4
Nitronic 60	
HR, CR; Rnd Rod	G7DC18AB1
<u>[H] High temperature steel</u>	
<u>Nickel Chromium</u>	
A286 (140 ksi)	
Plt & Sht; 600-800F	H1AB13AA15
Plt & Sht	H1AB13AB1
Forg, L-T, T-L, L-R	H1AB23AB1
A286 (160 ksi)	
Plt & Sht; 600-800F	H1AC13AA15
Plt & Sht	H1AC13AB1
Forg. rod, L-R	H1AC28AB1
A286 (200 ksi Bolt Material)	
Forg. rod, L-R	H1AD28AB1

JBK-75	
ST-CR-A; Plt, T-L	H1CB12AB1
[J] Tool Steel	
AISI Tool Steel	
M-50	
61-63 Rc; Plt	J1IK10AB1
T1(18-4-1)	
60-63 Rc; Plt	J1MA10AB1
[M] 1000-9000 Series aluminum	
2000 Series	
2014-T6	
Plt & Sht; L-T	M2AD11AB1
Plt & Sht; T-L	M2AD12AB1
2014-T651	
Plt & Sht; L-T	M2AF11AB1
Plt & Sht; T-L	M2AF12AB1
Plt & Sht; GTA Weld	M2AFB1AB1
Plt & Sht; GTA Weld, SR	M2AFB2AB1
2020-T651	
Plt & Sht; L-T	M2CB11AB1
Plt & Sht; T-L	M2CB12AB1
2024-T3	
Clad, Plt & Sht; L-T; LA & HHA	M2EA11AB1
Clad, Plt & Sht; T-L; LA & HHA	M2EA12AB1
Clad, Plt & Sht; L-T; DW	M2EA11WA1
Clad, Plt & Sht; T-L; DW	M2EA12WA1
2024-T351	
Plt & Sht; L-T; 300F to 400F Air	M2EB11AA11
Plt & Sht; L-T; LA & HHA	M2EB11AB1
Plt & Sht; T-L; LA & HHA	M2EB12AB1
2024-T3511	
Extr; L-T; LA & HHA	M2EC31AB1
2024-T62	
Plt & Sht; L-T; LA, HHA & ASW	M2EG11AB1
Plt & Sht; T-L; LA, HHA & ASW	M2EG12AB1
2024-T81	
Plt & Sht; L-T; 350F Air	M2EI11AA11
Plt & Sht; L-T	M2EI11AB1
Plt & Sht; L-T; DA	M2EI11AC1
Plt & Sht; L-T; HHA	M2EI11AD1
Plt & Sht; T-L; 350F Air	M2EI12AA11
Plt & Sht; T-L	M2EI12AB1
2024-T851	
Plt & Sht; T-L; 300F to 350F Air	M2EJ12AA11
Plt & Sht; L-T & T-L; LA, DA, JP-4	M2EJ13AB1
Plt & Sht; L-T & T-L 3.5% NaCl	M2EJ13WB1
2024-T852	
Forg; L-T & T-L, LA & DA	M2EK23AB1

2024-T861	
Plt & Sht; L-T; 300F to 400F Air	M2EL11AA11
Plt & Sht; L-T; LA & HHA	M2EL11AB1
Plt & Sht; T-L	M2EL12AB1
2048-T851	
Plt & Sht; L-T; LA, DA	M2FC11AB1
Plt & Sht; T-L; LA, DA	M2FC12AB1
2124-T851	
Plt & Sht; L-T; 120F to 350F Air	M2GC11AA10
Plt & Sht; L-T; LA, DA, HHA	M2GC11AB1
Plt & Sht; T-L; 300F - 400F Air	M2GC12AA11
Plt & Sht; T-L; LA, HHA	M2GC12AB1
Plt & Sht; T-L; -200F to -150F GN2	M2GC12GB6
Plt & Sht; S-T, S-L; LA, HHA	M2GC15AB1
2219-T62	
Plt & Sht; L-T	M2IA11AB1
Plt & Sht; L-T & T-L; 350F Air	M2IA13AA11
Plt & Sht; T-L	M2IA12AB1
Plt & Sht; L-T & T-L; -320F LN2	M2IA13LA4
2219-T851	
Plt & Sht; L-T, LA, DA	M2IC11AB1
Plt & Sht; T-L; LA, DA	M2IC12AB1
2219-T87	
Plt & Sht; L-T; 300F to 350F Air	M2IF11AA11
Plt & Sht; L-T	M2IF11AB1
Plt & Sht; L-T; -320F LN2	M2IF11LA4
Plt & Sht; T-L; 300F to 350F Air	M2IF12AA11
Plt & Sht; T-L	M2IF12AB1
Plt & Sht; T-L; -320F LN2	M2IF12LA4
Plt & Sht; GTA weld, PAR	M2IFB1AB1
Plt & Sht; GTA weld, PAR; -320F LN2	M2IFB1LA4
2324-T39	
Plt & Sht; L-T	M2JA11AB1
2090-T8E41	
Plt & Sht; L-T	M2PA11AB1
5000 Series	
5083-O	
Plt; T-L	M5BA12AB1
6000 Series	
6061-T6	
Plt; T-L	M6AB13AB1
Plt; GTA weld, PAR	M6ABA1AB1
6061-T651	
Plt; L-T & T-L 300F Air	M6AC13AA10
Plt; L-T & T-L	M6AC13AB1
6063-T5	
Plt & Sht; T-L; LA	M6BA12AB1

7000 Series	
7005-T6 & T63	
Plt & Sht; L-T	M7BA11AB1
Plt & Sht; T-L	M7BA12AB1
7010-T73651	
Plt & Sht; L-T & L-S	M7DA11AB1
7050-T73511	
Extr; L-T; LA, HHA, DA	M7GE31AB1
7050-T736 & T74	
Forg; L-T	M7GI21AB1
Forg; T-L	M7GI22AB1
7050-T73651 & T7451	
Plt & Sht; L-T; LA & HHA	M7GJ11AB1
Plt & Sht; L-T; DA	M7GJ11AC1
Plt & Sht; T-L; LA & HHA	M7GJ12AB1
Plt & Sht; T-L; DA	M7GJ12AC1
Plt & Sht; S-T	M7GJ15AB1
7050-T74511	
Extr; L-T; LA & DW	M7GL31AB1
7050-T73652 & T7452	
Forg; L-T	M7GM21AB1
Forg; T-L	M7GM22AB1
7050-T7651	
Plt & Sht; L-T; LA & HHA	M7GQ11AB1
Plt & Sht; T-L	M7GQ12AB1
7050-T76511	
Extr; L-T; LA & HHA	M7GS31AB1
Extr; T-L	M7GS32AB1
7000 Series	
7075-T6	
Plt, Sht & Clad;; L-T & T-L; LA	M7HA13AB1
Plt, Sht & Clad;; L-T & T-L; HHA	M7HA13AD1
7075-T651	
Plt & Sht; L-T; LA, DA	M7HB11AB1
Plt & Sht; L-T; HHA	M7HB11AD1
Plt & Sht; L-T; 3.5% NaCl	M7HB11WB1
Plt & Sht; T-L; DW	M7HB13WA1
Plt & Sht; S-T	M7HB15AB1
7075-T6510	
Extr; L-T; LA & DA	M7HC31AB1
7075-T6511	
Extr; L-T; LA & HHA	M7HD31AB1
Extr; T-L	M7HD32AB1
7075-T73	
Plt & Sht; L-T; LA, DA, HHA	M7HG11AB1
Plt & Sht; T-L	M7HG12AB1

7075-T7351	
Plt & Sht; L-T	M7HH11AB1
Plt & Sht; L-T; DA	M7HH11AC1
Plt & Sht; L-T; HHA	M7HH11AD1
Plt & Sht; T-L; LA & DA	M7HH12AB1
Plt & Sht; S-T; LA	M7HH15AB1
7075-T73510	
Extr; L-T; LA	M7HI31AB1
7075-T73511	
Extr; L-T; LA, DA, HHA	M7HJ31AB1
7075-T7352	
Plt, Sht & Forg; L-T; LA & DA	M7HK11AB1
Plt, Sht & Forg; T-L; LA & DA	M7HK12AB1
7075-T7651	
Plt & Sht; L-T; LA & DA	M7HM11AB1
Plt & Sht; T-L; LA & DA	M7HM12AB1
7079-T651	
Plt & Sht; L-T	M7IC11AB1
7000 Series	
7149-T73511	
Extr; L-T; LA	M7NA31AB1
Extr; T-L; LA	M7NA32AB1
7178-T6 & T651	
Plt & Sht; L-T; LA & HHA	M7RA11AB1
Plt & Sht; T-L; LA & HHA	M7RA12AB1
7178-T7651	
Plt & Sht; L-T	M7RF11AB1
Plt & Sht; T-L	M7RF12AB1
7475-T61	
Plt, Sht, Clad; L-T; LA, DA, HHA	M7TB11AB1
Plt, Sht, Clad; T-L; LA, DA, HHA	M7TB12AB1
7475-T651	
Plt & Sht; L-T; LA, DA, HHA	M7TD11AB1
7475-T7351	
Plt & Sht; L-T; LA, DA, HHA, DW	M7TF11AB1
Plt & Sht; T-L; LA, DA, HHA	M7TF12AB1
7475-T7651	
Plt & Sht; L-T; LA, DA, HHA, DW, 3.5%NaCl	M7TJ11AB1
[O] Misc. and cast aluminum	
300 Series cast	
A356-T60	
Cast	O3FB50AB1
[P] Titanium alloys	
Ti Unalloyed	
Ti-55	
Plt & Sht	P1AA13AB1
Plt & Sht; DW & SW	P1AA13WA1

Ti-70	
Plt & Sht	P1CA13AB1
Plt & Sht; DW & SW	P1CA13WA1
Binary Alloys	
Ti-2.5 Cu; STA	
Sht; LA, HHA, DW	P2AA13AB1
Ternary Alloys	
Ti-5Al-2.5Sn; Annealed	
Sht; LA, HHA, DW	P3CA13AB1
Ti-5Al-2.5Sn (ELI); Annealed	
Forg	P3CB23AB1
Forg; -423F LH2	P3CB23LA3
Ti-3Al-2.5V; CW, SR(750F)	
Extr	P3DB33AB1
Ti-6Al-4V(MA)	
Plt & Sht, -100F	P3EA13AA7
Plt & Sht	P3EA13AB1
Forg	P3EA23AB1
Extr	P3EA33AB1
Ti-6Al-4V; BA(1900F/.5h + 1325F/2h)	
Plt & Sht; LA, DA, 3.5% NaCl	P3EB12AB1
Forg; LA, DA, HHA, 3.5% NaCl	P3EB23AB1
Ti-6Al-4V; RA	
Sht; L-T; LA,DA,HHA,DW, 3.5% NaCl	P3EC11AB1
Sht; T-L; LA,DA,HHA,DW, 3.5% NaCl	P3EC12AB1
Plt; -100F	P3EC13AA7
Plt; LA, DA, HHA, DW	P3EC13AB1
Forg; LA, DA, HHA, 3.5% NaCl	P3EC23AB1
Ti-6Al-4V; ST(1750F) + A(1000F/4h)	
Plt & Sht; SR(1000F/4h)	P3ED13AA1
Plt & Sht; SR(1000F/8h)	P3ED13AB1
Plt & Sht; SR(1000F/4h); -320F LN2	P3ED13LA4
Forg; SR(1000F/4h)	P3ED20AB1
Forg; SR(1000F/4h); -320F LN2	P3ED20LA4
GTA Weld; SR; thk < 0.2"	P3EDB2AB1A
GTA Weld; SR; thk >= 0.2"	P3EDB2AB1B
Ti-6Al-4V; ELI; BA(1900F/.5h) + 1325F/2h)	
Plt & sht; LA, 3.5% NaCl	P3EL12AB1
Ti-6Al-4V (ELI) RA	
Plt	P3EM13AB1
Forg; -100F	P3EM23AA7
Forg	P3EM23AB1
Forg; -452F LHe	P3EM23LA2
Forg; -320F LN2	P3EM23LA4
Forg; EB welded, SR; weldline	P3EMD2AB1
Forg; EB welded, SR; weldline; -320F LN2	P3EMD2LA4
Forg; EB welded, SR; HAZ	P3EMD8AB1
Forg; EB welded, SR; HAZ; -320F LN2	P3EMD8LA4

<u>Quaternary Alloys</u>	
Ti-4.5Al-5Mo-1.5Cr	
Plt; LA, 3.5% NaCl	P4BA10AB1
Ti-8Al-1Mo-1V	
Sht	P4CB11AB1
Ti-6Al-6V-2Sn MA	
Plt, Forg, Extr; LA, DA, HHA, DW	P4DA33AB1
Ti-6Al-6V-2Sn RA	
Plt	P4DB12AB1
Ti-6Al-6V-2Sn BA	
Plt	P4DC12AB1
Ti-6Al-6V-2Sn ST(1600F); A(1000F/6h)	
Forg; -65F	P4DD21AA7
Forg; 300F	P4DD21AA10
Forg; LA, DA, HHA	P4DD21AB1
Ti-10V-2Fe-3Al	
STA(140-160 UTS, 70Klc) Plt & Forg	P4MD23AB1
STA(160-180 UTS, 60Klc) Plt & Forg	P4MF23AB1
STA(180-200 UTS, 40Klc) Plt & Forg	P4MG13AB1
STA(180-200 UTS, 30Klc) Plt & Forg	P4MG20AB1
STA(180-200 UTS, 25Klc) Forg	P4MG23AB1
Ti-6Al-2Zn-2Sn-2Mo-2Cr (ST or STA)	
Plt; HHA	P5FB11AD1
<u>[Q] Ni alloys/superalloys</u>	
<u>Hastelloy Alloys</u>	
Hastelloy B	
Rnd Rod	Q1AA16AB1
Hastelloy X-280; ST(2150F)	
Plt; 600-800F Air	Q1QA10AA15
Plt; 1000-1200F Air; >.67Hz	Q1QA10AA19
Plt	Q1QA10AB1
<u>Inconel Alloys</u>	
Inconel 600	
Plt & Sht; 1000F	Q3AB10AA18
Plt & Sht; 75-800F	Q3AB10AB1
Inconel 625	
Plt & Sht; 600F	Q3EA10AA14
Plt & Sht; 800F	Q3EA10AA16
Plt & Sht; 1000F	Q3EA10AA18
Plt & Sht;	Q3EA10AB1
Inconel 706; ST(1800-1950F); A(1375F/8h; 1150F/5-8h)	
Forg & Extr	Q3JB33AB1
Forg & Extr; -452F Lhe	Q3JB33LA2
ST Plt - GTA weld - STA	Q3JBB3AB1
ST Plt - GTA weld - STA; -452F Lhe	Q3JBB3LA2

Inconel 718; ST(1700-1850F) + A(1325F/8h + 1150F/10h)	
Plt; 600F air, >.3Hz	Q3LB11AA14
Plt; 800F air, >.3Hz	Q3LB11AA16
Plt; 1000F air, >.3Hz	Q3LB13AA18
Sht (t<.25")	Q3LB13AB1A
Plt	Q3LB13AB1B
Forg	Q3LB23AB1
Forg; 300F air, >.3Hz	Q3LB26AA10
Forg; 600F air, >.3Hz	Q3LB26AA14
Forg; 800F air, >.3Hz	Q3LB26AA16
Forg; 1000F air, >.3Hz	Q3LB26AA18
GTA weld-STA; 600F air, >.6Hz	Q3LBB3AA14
GTA weld-STA; 800F air, >.6Hz	Q3LBB3AA16
GTA weld-STA; 1000F air, >.6Hz	Q3LBB3AA18
ST plt-GTA weld-aged	Q3LBB3AB1
ST plt-GTA weld-aged; -320F LN2	Q3LBB3LA4
ST plt-EB weld-aged	Q3LBD3AB1
ST plt-EB weld-aged; -320F LN2	Q3LBD3LA4
Inconel 718; ST(1900F) + A(1400F/10h + 1200F/10h)	
Plt; -320F LN2	Q3LC10LA4
Plt; 600F air; >.6Hz	Q3LC11AA14
Plt	Q3LC11AB1
Plt; 1000F air; >.6Hz	Q3LC12AA18
Inconel 718; ST(2000F) + A(1325F/4h + 1150F/16h)	
Plt; 800F air; >.2Hz	Q3LE11AA16
Plt; 1000F air; >.6Hz	Q3LE13AA18
Plt	Q3LE13AB1
GTA weld-STA; 600F air, >.6Hz	Q3LEB3AA14
GTA weld-STA; 800F air, >.6Hz	Q3LEB3AA16
GTA weld-STA; 1000F air, >.6Hz	Q3LEB3AA18
GTA weld-STA	Q3LEB3AB1
Inconel 718, Bolt material	
185 ksi UTS Bolts	Q3LP18AB1
225 ksi UTS Bolts	Q3LQ18AB1
Inconel X-750; ST(2100F) + A(1550F/24h + 1300F/20h)	
Plt; 600F air, >.6Hz	Q3SD10AA14
Plt; 800F air, >.6Hz	Q3SD10AA16
Plt; 1000F air, >.6Hz	Q3SD10AA18
Plt & Forg	Q3SD26AB1
Forg; -452F Lhe	Q3SD26LA2
Rene and Udimet Alloys	
Rene 41; ST(1950F) + A(1400F/16h)	
Plt & Forg	Q7AD13AB1
Forg; 1100F air	Q7AD26AA19
Forg; 1200F air	Q7AD26AA20

[R] Misc. superalloys	
<u>Multiphase alloys</u>	
MP35N Rnd Rod	R3AB18AB1
[S] Copper/Bronze alloys	
<u>Be-Cu Alloys</u>	
CDA 172	
Rnd Rod	S0BA13AB1
C17510	
Peak-aged Plt	S1LB11AB1
Peak-aged Plt; -320F LN2	S1LB11LA4
Overaged Plt	S1LC11AB1
<u>Al-Bronze Alloys</u>	
CDA 630 Al-Bronze Extr	S6JB36AB1
[T] Magnesium alloys	
AM 503 Plt	T1AA11AB1
AZ-31B-H24 Plt	T1DA12AB1
ZK-60A-T5 Plt	T1MA12AB1
ZW1 Plt	T1NA11AB1
QE22A-T6 Plt	T2LB13AB1
[U] Misc. non-ferrous alloys	
<u>Beryllium</u>	
Cross-rolled sht	U1CA90AB1
Hot-pressed blk	U1CA93AB1
<u>Columbium Alloys</u>	
C-103 Plt	U2CA10AB1
<u>Zinc Alloys</u>	
Zn-4Al-0.04Mg Die cast alloy No. 3	
As cast; 135-200F air0	U4BA50AA9
As cast	U4BA50AB1